LENGUAJE MAQUINA DEL ZX SPECTRUM

SUBRUTINAS Y TRUCOS

P. PELLIER

LENGUAJE MAQUINA DEL ZX SPECTRUM

Editorial Gustavo Gili, S. A.

08029 Barcelona Rosellón, 87-89. Tel. 322 81 61
28006 Madrid Alcántara, 21. Tel. 401 17 02
1064 Buenos Aires Cochabamba, 154-158. Tel. 361 99 98
03100 México, D.F. Amores, 2027. Tels. 524 03 81 y 524 01 35
Bogotá Diagonal 45 N.º 16 B-11. Tel. 245 67 60
Santiago de Chile Santa Victoria, 151. Tel. 222 45 67

LENGUAJE MAQUINA DEL ZX SPECTRUM

SUBRUTINAS Y TRUCOS

P. PELLIER

GG

Esta obra es la traducción del libro francés
Langage Machine, Trucs et Astuces sur ZX Spectrum,
de Pellier, publicado por Éditions Eyrolles, de París.

Versión castellana de Joan Pelegrín Muniente

Printed in Spain
ISBN: 84.252-1207-3
Depósito legal: B. 9.127-1985
FOTOCOMPOSICION: TECFA, S. A., Barcelona
IMPRESION: HUROPE, S. A. - Recaredo, 2 - Barcelona

Indice

Advertencia

Esta obra se dirige a todos aquellos que desean iniciarse en la programación en ensamblador del Z 8Ø y en su funcionamiento sobre el ZX SPECTRUM. Para la lectura de este libro son necesarios unos conocimientos básicos sobre informática y algunas nociones del Basic, ya que las dificultades serán progresivas.

El primer capítulo describe el funcionamiento del microprocesador Z 8Ø y detalla el juego de instrucciones de que dispone. Está lleno de reflexiones sobre el interés y los peligros de utilización de cada típo de instrucciones sobre el ZX SPECTRUM.

El segundo capítulo se acerca paso a paso a la utilización de los programas de utilidad del ensamblador, que son el Editor/Ensamblador y el Debugger.

El tercer capítulo ofrece cierto número de subprogramas que serán excelentes ejemplos de programas para principiantes.

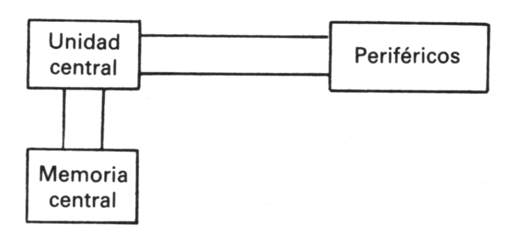
El capítulo último expone los procedimientos de diálogo en ensamblador entre el ZX SPECTRUM y cada uno de los órganos periféricos que son: la pantalla de visualización, la impresora, la interfase sonora, el teclado, los mandos de juego y el magnetófono. Este capítulo explicará cómo escribir o dibujar sobre la pantalla, leer el teclado para conocer las teclas pulsadas, manejar el mando de juego y la interfase sonora.

Finalmente, en los anexos 2 y 3 encontrará dos listas exhaustivas de instrucciones sobre el Z 8Ø, clasificadas de dos maneras distintas.

1. La programación en ensamblador

1.1. Introducción: estructura interna del microordenador

Un sistema microordenador consta de tres elementos fundamentales que son: la unidad central, la memoria central y los periféricos. Estos elementos están entrelazados entre ellos por buses, los cuales están constituidos por cierto número de conexiones eléctricas destinadas a permitir la transferencia de informaciones.



La unidad central es el núcleo del microordenador; ella es quien controla la transferencia de datos con la memoria central y con los periféricos.

El componente fundamental de la unidad central es el microprocesador. Es el que realiza la mayor parte de las funciones de la unidad central, o sea los cálculos. En el SPECTRUM, este microprocesador es fabricado por ZILOG y se llama Z 8Ø.

La memoria central es un elemento esencial del microordenador. Sin ella éste no podría funcionar. Está destinada a contener los programas ejecutados y los datos manipulados por la unidad central.

1.1.1. La memoria central

La memoria central está dividida en cierta cantidad de posiciones pudiendo contener un dato numérico entero comprendido entre Ø y 255. Cada una de estas posiciones tiene 8 bits y se llama *octeto* o byte en inglés. El bit es el elemento más pequeño de la memoria. Solamen-

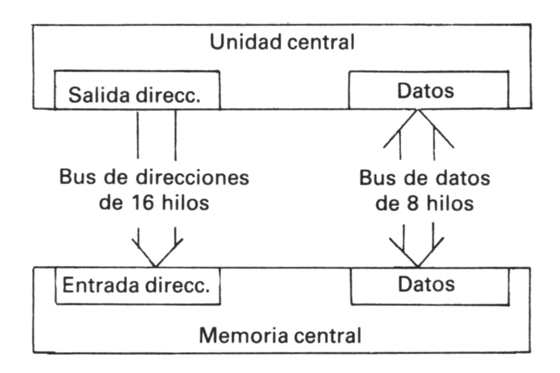
te puede tomar dos valores: Ø (estado bajo: Ø V) y 1 (estado alto: 5 V), lo cual permite memorizar un número entero comprendido entre Ø y 1. Agrupando 8 bits se obtiene un octeto que puede memorizar un número comprendido entre ØØØØ ØØØØ y 1111 1111 en binario,* o sea Ø y 255 en decimal. Se ha tomado el hábito de utilizar la base 16 (representación hexadecimal) para indicar el valor de un octeto. Con esta base el valor del octeto varía desde ØØ a FF. Esta representación corresponde lo mejor posible al reparto en octetos, ya que permite codificar un octeto sólo mediante dos símbolos, y cualquier número hexadecimal de dos cifras puede ser colocado en un octeto.

De manera simbólica podemos decir que la memoria está constituida por una cantidad de octetos colocados uno detrás de otro formando cadena. A cada octeto se le asocia un número de orden que representa su posición dentro de la cadena. A este número se le llama *dirección* del octeto. El primero tendrá la dirección Ø, el segundo la dirección 1 y así sucesivamente hasta el último. Esta dirección servirá para seleccionar cualquiera de los octetos de la memoria central.

El microprocesador Z 8Ø es capaz de manipular direcciones de memoria comprendidas entre Ø y 65535 ($2^{16} - 1$). Para ello dispone de un conjunto de 16 salidas de dirección. Estas salidas sólo pueden tener los valores Ø y 1; gracias a ello se pueden formar todas las combinaciones de números binarios comprendidos entre ØØØØ ØØØØ ØØØØ ØØØØ y 1111 1111 1111 1111, o sea Ø y 65535 en decimal, o ØØØØ y FFFF en hexadecimal. Estas 16 salidas están ligadas a 16 entradas de dirección de la memoria central a través del bus de direcciones. Aplicando una combinación binaria a estas salidas de dirección, el microprocesador puede seleccionar uno de los octetos de la memoria central. Para utilizar este octeto el microprocesador tiene 8 bornes de datos conectados a 8 bornes de datos de la memoria central mediante el bus de datos. Así, el octeto de la memoria central seleccionado por el microprocesador podrá transitar entre la memoria central y el microprocesador. Los bornes de datos y el bus de datos son bidireccionales para permitir la transferencia de octetos entre el microprocesador y la memoria (escritura en memoria), o bien entre la memoria y el microprocesador (lectura en memoria).

Existen dos tipos de memoria en el SPECTRUM. El primero, la memoria muerta (en inglés ROM = Read Only Memory), que solamente funciona en lectura. Su contenido es memorizado una sola vez por todas; no puede ser modificado por el microprocesador. Su valor es de 16 384 octetos. Para indicar este valor generalmente se utiliza el kiloocteto, que es igual a 2^{10} octetos, o sea 1 Ø24. El valor de la memoria muerta (ROM) es pues de 16 K (abreviación de kiloocteto).

*En el anexo 1 se da una explicación sobre las bases de numeración.



El segundo tipo de memoria es la viva (en inglés RAM = Random Access Memory), que trabaja tanto en lectura como en escritura. Los datos registrados en esta memoria sólo son memorizados cuando el microordenador está alimentado por corriente eléctrica. En el SPECTRUM, el valor de esta memoria es de 16 K o 48 K, según la configuración del modelo.

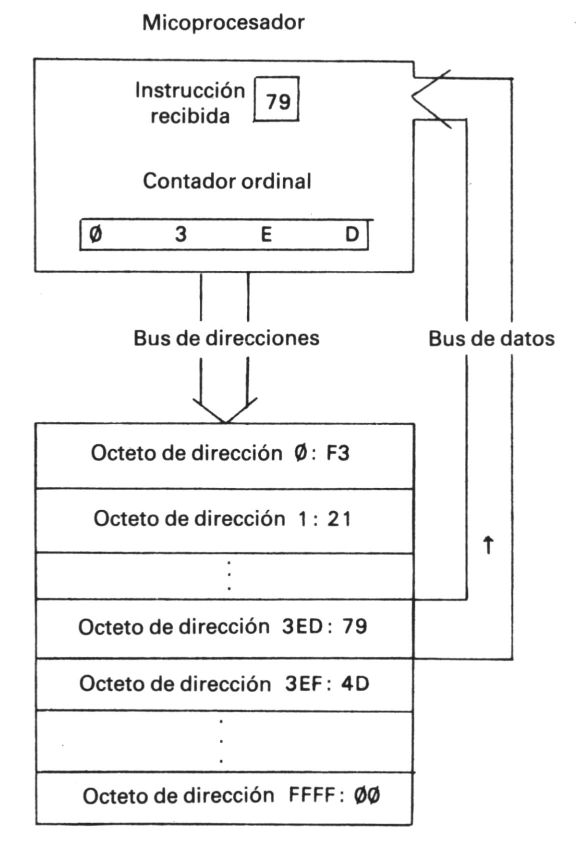
1.1.2. El microprocesador

El microprocesador Z 8∅ es una máquina capaz de ejecutar cierta cantidad de *instrucciones*. Estas se distinguen mediante un código, el cual se introduce en 1, 2, 3 o 4 octetos. Las operaciones ordenadas por estas instrucciones generalmente son muy simples (adición, sustracción, operación lógica, transferencia de memoria).

El microprocesador ha sido concebido para ejecutar secuencialmente una serie de instrucciones almacenadas en la memoria central, ejecutando así un programa almacenado en la memoria. Para ello dispone de una memoria interna de 16 bits llamada contador ordinal. Este contador ordinal sirve para memorizar la dirección de la próxima instrucción a ejecutar. El ciclo completo para ejecutar una instrucción es el siguiente.

1. El contador ordinal se encuentra ligado a las salidas de dirección para seleccionar el octeto que contiene el código de la instrucción a ejecutar.
2. El octeto seleccionado llega al microprocesador por el bus de datos.
3. El microprocesador descodifica la instrucción y ejecuta la acción correspondiente.
4. El contenido del contador ordinal queda aumentado en el número de octetos de la instrucción ejecutada, de manera que contenga la dirección de la instrucción siguiente.

5. El proceso se repite a partir de la etapa 1 para la instrucción siguiente.



La instrucción situada en la dirección 3ED (valor del contador ordinal) es transferida hacia el Z 8Ø que la ejecutará. Aumentará en uno el contador ordinal y ejecutará la instrucción siguiente situada en la dirección 3EE (hexadecimal).

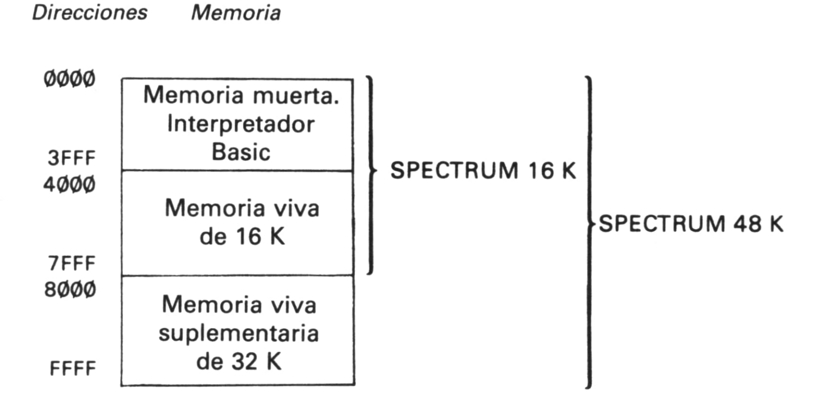
De esta manera, el microprocesador ejecuta cada instrucción del programa guardado en la memoria central con la forma de códigos binarios. Así pues, puede decirse que es un programa escrito en *lenguaje máquina* (o código máquina) que es el único lenguaje capaz de interpretar el microprocesador.

El lenguaje Basic es totalmente incomprensible para el microprocesador que no sabe ejecutar una orden como PRINT. Para la utilización del lenguaje Basic en el microordenador ha sido necesario crear una interfase entre el lenguaje binario manipulado por el microprocesador y las órdenes del lenguaje Basic. Este lenguaje, escrito naturalmente en lenguaje máquina, y que es el único que puede comprender

el microprocesador, está colocado en los 16 K de memoria muerta. De esta forma está constantemente en las memorias del ordenador. Cuando se conecta a tensión se activa automáticamente por la puesta a cero del contador ordinal. El interpretador, que empieza en la dirección cero, es así puesto en marcha al conectar la tensión. El usuario tiene entonces la impresión de trabajar sobre una máquina que solamente comprende el lenguaje Basic.

Todo programa escrito en lenguaje máquina (como el interpretador Basic) tiene necesidad de manipular cierto número de datos. Debido a ello se utiliza la memoria viva para escribir, leer o modificar las variables que utiliza. Estas variables están constituidas por octetos de la memoria central, y pueden transitar entre el microprocesador y la memoria. El microprocesador, que no contiene más que un bus de datos de 8 hilos, no puede leer más de un octeto a la vez. Así, se dice que es un *microprocesador de 8 bits*. Para manipular variables que contengan varios octetos será necesario descomponerlas. Descubrimos así el interés de los microprocesadores de 16 bits y de 32 bits, que permiten manipular de una sola vez variables de tamaño más grande. Esto comporta más rapidez y facilidades en el desarrollo de los programas.

Todo programa en lenguaje máquina, que no sea el interpretador del Basic, deberá colocarse en la memoria viva. Demos por este motivo la distribución de la ocupación de las memorias (en inglés: memory map) en el ZX SPECTRUM.



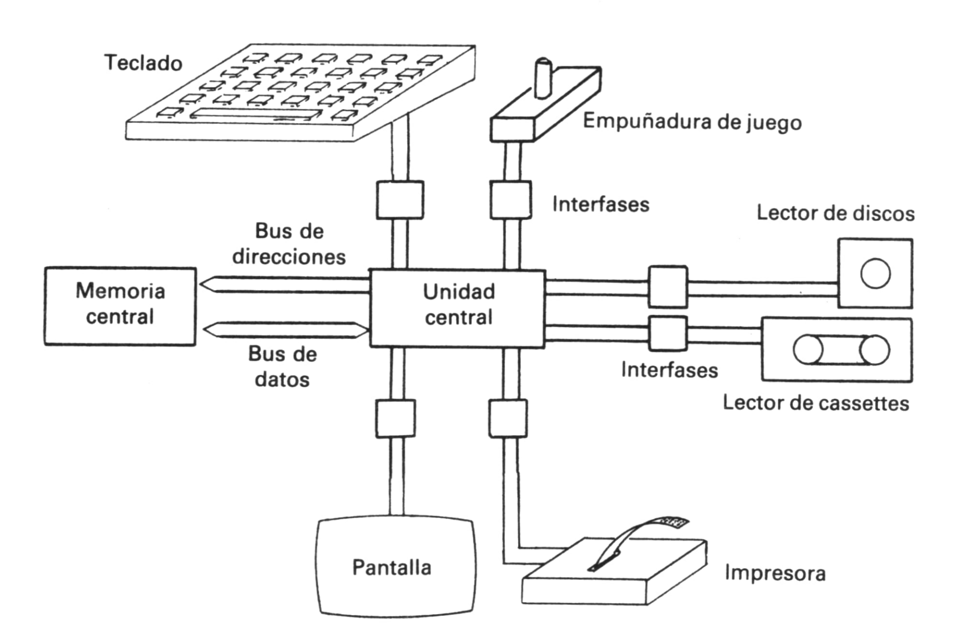
1.1.3. Los periféricos

Los periféricos son los órganos de enlace entre el ordenador y el mundo exterior. No son primordialmente necesarios para el funcionamiento del ordenador, pero sin ellos no sirve para nada, pues por sí solo no puede comunicar sus resultados al usuario.

Existen principalmente tres tipos de periféricos:

— Los periféricos que funcionan como entradas y que recogen las órdenes del usuario. Como es el caso del teclado y los mandos de juego.
— Los periféricos que funcionan como salidas y sirven para visualizar los resultados obtenidos por el ordenador. Como son la pantalla de video y la impresora.
— Los periféricos que funcionan como entradas y salidas; son el lector de cassettes y las unidades de disco. El intercambio de informaciones entre el periférico y el ordenador se realiza así en forma bidireccional.

Para permitir la utilización de periféricos, el microordenador contiene interfases materiales que aseguran la conexión entre la unidad central, el bus de datos, el bus de direcciones y los periféricos. Algunos de los subprogramas escritos en lenguaje máquina introducidos en la ROM permiten la gestión software de los periféricos. Dichos programas son calificados de «*rutinas de gestión de las entradas/salidas*».



1.2. Ventajas e inconvenientes del lenguaje máquina

El lenguaje máquina presenta dos ventajas importantes con respecto a un lenguaje de alto nivel como es el Basic.

Es el lenguaje más rápido que puede encontrarse para un microordenador, ya que puede suministrar directamente al microprocesador los códigos de las instrucciones a ejecutar, sin pasar por una interfase software como el interpretador del Basic. La diferencia de velocidad entre estos dos lenguajes es considerable (el lenguaje máquina va alrededor de 1ØØ veces más rápido). La diferencia es menos acentuada con los lenguajes compilados, como son el Pascal o el Basic compilado. Estos son más rápidos que los lenguajes interpretados, porque el programa escrito en lenguaje de alto nivel se traduce una sola vez a lenguaje máquina por un procedimiento llamado «compilación». El programa traducido podrá ser ejecutado directamente por el microprocesador. Por el contrario, los interpretadores no traducen las instrucciones del lenguaje de alto nivel, sino más bien simulan su funcionamiento durante la ejecución, haciendo uso de las posibilidades del microprocesador. Si los compiladores fuesen perfectos producirían el código máquina más rápido posible para un programa determinado en lenguaje de alto nivel. En este caso sería inútil programar directamente en lenguaje máquina. De hecho, los compiladores producen un código máquina poco optimizado, que es varias veces más lento que el mismo programa realizado directamente en lenguaje máquina y que utiliza mejor las posibilidades del microprocesador.

Además de la rapidez, el lenguaje máquina permite acceder a todas las posibilidades del microordenador y escapar así de las restricciones impuestas por el lenguaje de alto nivel. Con el ensamblador se podrá programar la salida sonora para producir algo distinto al «bip» clásico, utilizar las dos últimas líneas de la pantalla para un fin distinto al de recoger información debido a la orden de INPUT, o utilizar una cantidad mayor de caracteres gráficos programados.

Si el lenguaje máquina solamente presentara estas ventajas, nos preguntaríamos porqué los constructores de ordenadores se empeñan en implantar el Basic en la versión de base de su microordenador. De hecho, el lenguaje máquina es mucho más difícil de utilizar para un neófito. El mínimo programa de cálculo aritmético con registro de resultados sobre la pantalla que se escribe en unas pocas líneas Basic, necesita varios centenares de instrucciones en lenguaje máquina, simplemente porque el microprocesador no sabe hacer otra cosa que sumas, restas y operaciones lógicas sobre números enteros codificados sobre 8 o 16 bits.

1.3. El ensamblador

La programación en código máquina es dura y ardua; debido a ello se ha creado *el lenguaje ensamblador* que asocia a cada instrucción de máquina una sucesión de caracteres alfanuméricos formando una palabra que recuerda de forma mnemotécnica la operación realizada por la instrucción. Se le llama *palabra mnemotécnica*.

Pongamos por ejemplo la operación: negación (cambio de signo). El código de esta instrucción es:

ED 44 (consta de dos octetos)

Su mnemotécnico es:

NEG

Para el programador, el mnemotécnico NEG es mucho más fácil de retener que el código hexadecimal ED 44, mucho más si tenemos en cuenta que hay centenares de instrucciones en lenguaje máquina (696 instrucciones en el Z 8Ø).

Los mnemotécnicos no son directamente ejecutables por el microprocesador. Se ha debido crear una interfase software que asegure la transcripción de los mnemotécnicos a códigos máquina (ensamblaje). Esta interfase es un programa generalmente escrito en lenguaje máquina al que se le llama *Ensamblador*. Además de la transcripción de mnemotécnicos a códigos máquina, el ensamblador aporta facilidades para la edición de programas en lenguaje ensamblador, facilidades para manipular las variables y los datos, así como la posibilidad de especificar las direcciones de bifurcación en las instrucciones de salto.

El hecho de que la puesta a punto de los programas en lenguaje máquina suele ser más difícil que la de los programas en Basic, ha motivado la creación de programas potentes para la puesta a punto. A estos programas se les llama en inglés *Debugger*. Generalmente son capaces de efectuar el desensamblaje de las instrucciones correspondientes (operación inversa del ensamblaje). A menudo poseen una orden para ejecutar las instrucciones paso a paso, lo que es muy práctico para la puesta a punto y para la comprensión del funcionamiento de las instrucciones por el principiante.

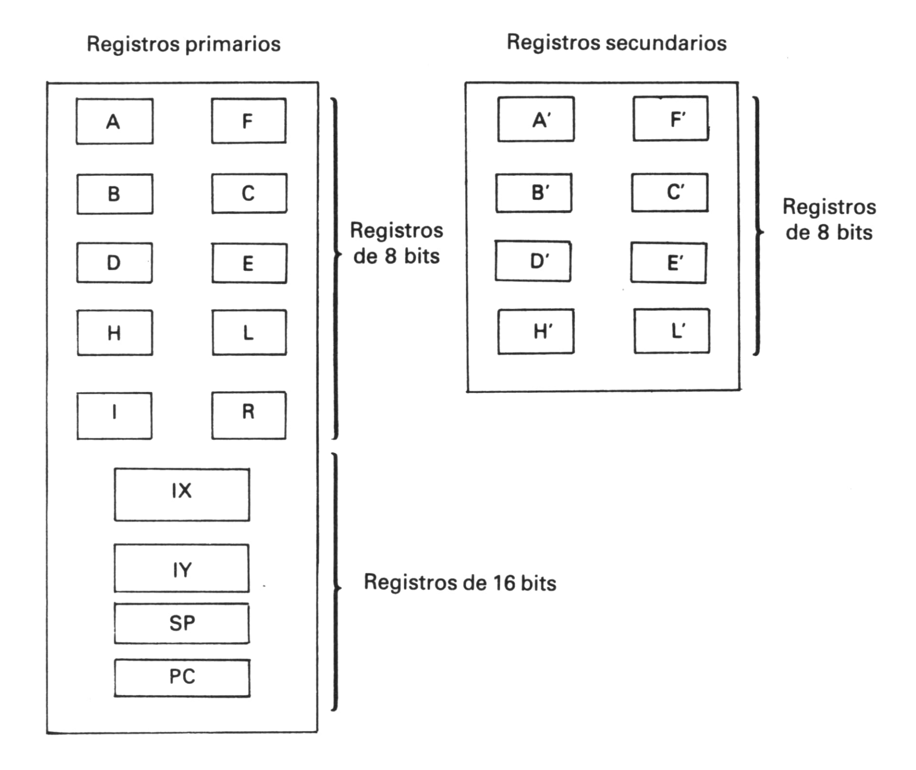
1.4. Los registros del Z 8Ø

Para trabajar, el microprocesador dispone de una cantidad de registros, que son memorias de trabajo ultrarrápidas, situadas en su interior. El Z 8Ø posee instrucciones para hacer operaciones en los re-

gistros y otras para realizar transferencias entre registros o bien entre registro y la memoria central. A *grosso modo*, un programa en lenguaje máquina utilizará la siguiente estructura:

— Transferencia de datos desde la memoria viva hacia los registros.
— Manipulación de los datos memorizados en los registros.
— Transferencia de los resultados contenidos en los registros hacia la memoria central.

El Z 8∅ contiene en total 22 registros y cada uno tiene utilizaciones particulares.



Los registros A, B, C, D, E, H y L son registros de 8 bits para usos generales. Se utilizan para manipular o memorizar temporalmente un dato representado por un octeto.

El registro A tiene una función particular. Todas las operaciones lógicas o aritméticas de 8 bits se efectuarán entre el registro A y otro registro o un octeto de la memoria. El resultado siempre se coloca en el registro A que ha recibido el nombre de *acumulador*.

El registro B a menudo se utiliza como contador de bucle en un programa. La instrucción DJNZ, que se parece a la orden NEXT del Basic, utiliza este registro como variable de bucle.

Los registros B y C, que son dos registros de 8 bits, pueden agruparse para formar un registro de 16 bits al que se llama BC. El registro B contiene los 8 bits más significativos del registro BC.

Registro B 3F   A7 Registro C

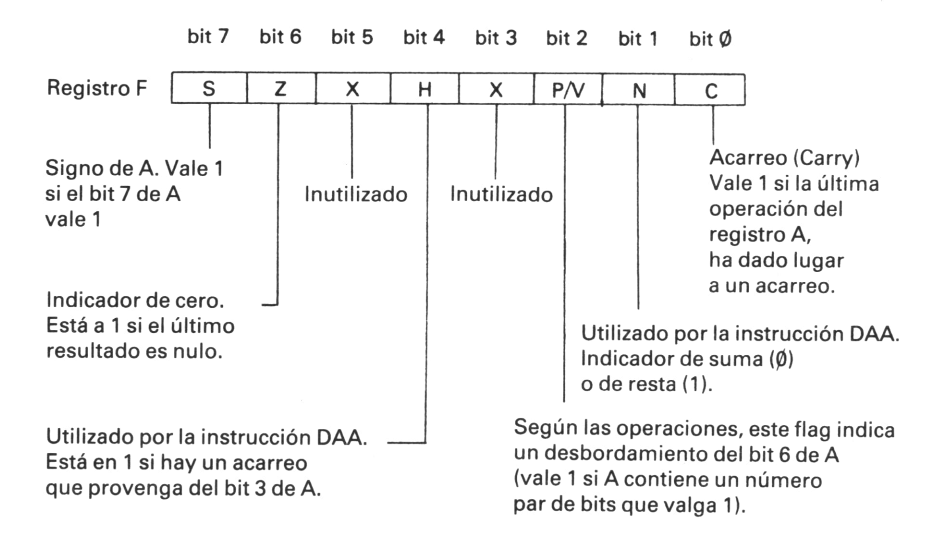
Registro BC 3FA7

Lo mismo ocurre con los registros D,E y H,L que forman respectivamente los registros dobles DE y HL.

El registro HL trabaja como acumulador para las operaciones aritméticas. Las sumas y restas se realizan entre este registro y uno de los registros BC o DE. El resultado se coloca en HL.

Observemos que por este procedimiento de agrupación de registros, el microprocesador Z 8Ø puede manipular registros de 16 bits. El Z 8Ø, de hecho, es un pseudomicroprocesador de 16 bits con bus de datos de 8 bits solamente. Esta característica lo hace más potente frente a los otros microprocesadores de 8 bits.

El registro F tiene 6 *indicadores* o *banderas* (flags en inglés). Cada indicador está hecho de un bit que se encuentra o bien en el estado VERDADERO (1), o bien en el estado FALSO (Ø). El cuadro siguiente resume los flags del registro F.



La mayor parte de estos indicadores informan al programa del estado del registro A después de una operación lógica o aritmética. Las instrucciones de salto condicional permiten comprobar el estado de un flag y bifurcar o no a una dirección de la memoria central según el estado del flag. Estas instrucciones de salto condicional asociadas al registro F, son el equivalente de la orden IF ... THEN ... del Basic.

Ciertas instrucciones modifican los flags según criterios distintos del estado del registro A. Otras no las modifican, pero sí provocan una modificación del registro A. Así pues, al usuario le corresponde asegurar que para cada una de las comprobaciones que efectúe, los flags contengan exactamente el valor deseado. Esta es una de las desventajas del lenguaje máquina, ya que es fuente de numerosos errores para los principiantes.

En la práctica se comprueba que, esencialmente, se utilizan los flags de cero (Z) y acarreo (C de carry) y menos a menudo el indicador de signo (S). Veremos con más detalle la utilización de los flags cuando examinemos el juego de instrucciones.

Los registros secundarios (A', F', B', C', D', E', H', L') funcionan de la misma forma que sus homólogos primarios (A, F, B, C, D, E, H, L). Estos permiten aumentar la capacidad de memorización del microprocesador. No obstante, tienen un interés limitado por el hecho de que el microprocesador no puede manipular simultáneamente los registros primarios y los registros secundarios. El Z 8Ø no puede llegar más que a los registros primarios, pero posee dos instrucciones (EX, AF, AF' y EXX) para cambiar de un bloque dos registros primarios con los registros secundarios.

El registro I, que estudiaremos con más detalle a continuación, se utiliza exclusivamente para gestionar las interrupciones.

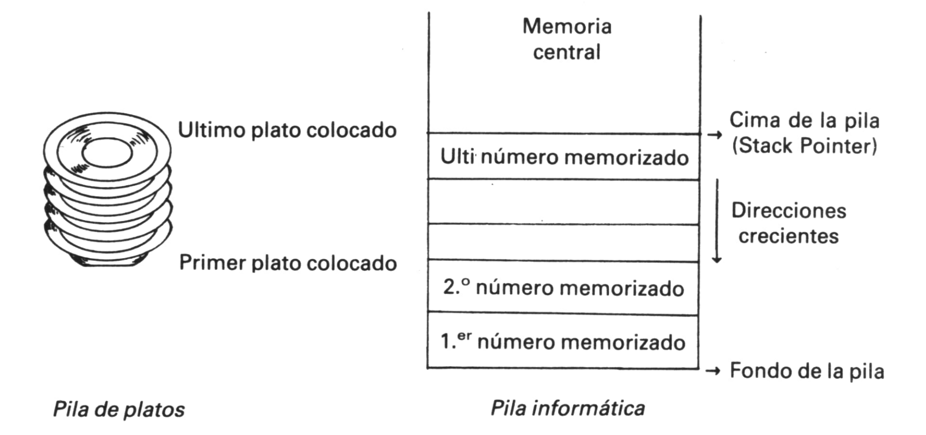
El registro R es un contador utilizado para refrescar las memorias dinámicas. Está controlado por el Z 8Ø y no es utilizable como registro por el usuario. El refresco de las memorias consiste en reescribir periódicamente el contenido de las memorias, ya que tienden a perder la información que contienen a causa de su tecnología dinámica. El registro R, utilizado como contador, permite saber cuándo debe ser ordenado el proceso de refresco. Su valor es aleatorio y no podrá ser utilizado más que para esto (generador de números aleatorios).

Los cuatro últimos registros (IX, IY, SP y PC), son registros de 16 bits que no pueden descomponerse en dos registros de 8 bits.

Los registros IX e IY son registros de índice. Son utilizados para permitir el direccionado por índice que estudiaremos en el párrafo siguiente. No obstante, pueden ser empleados como registros clásicos de 16 bits para memorizar un valor o realizar operaciones aritméticas.

El registro SP es *el puntero de la pila* (abreviación de stack pointer). La pila es un artificio software creado para poder utilizar subprogramas y para la salvaguarda temporal de registros. Su funcionamiento es análogo al de una pila de platos. En una pila de platos solamente puede retirarse fácilmente el plato de encima y no puede colocarse otro plato más que encima de la misma. Esta pila sigue la regla de «último en llegar, primero en salir» (en inglés LIFO: last in, first out). El último plato colocado es el primero en retirarse.

En informática la pila (stack) es parecida a una pila donde los platos son sustituidos por números de 16 bits. El lugar donde van a ser colocados estos números es la memoria central. La cima de la pila está simbolizada por el registro SP que contiene la dirección de memoria del último número registrado.

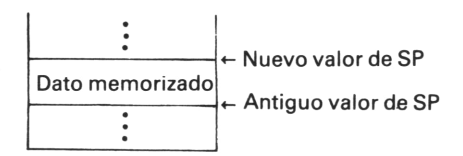


Pila de platos *Pila informática*

La operación de colocar un dato sobre la cima del stack se llama *apilamiento*. Esto se resume por las operaciones siguientes:

— SP ← SP − 2: decrementar el stack pointer de pila en 2.

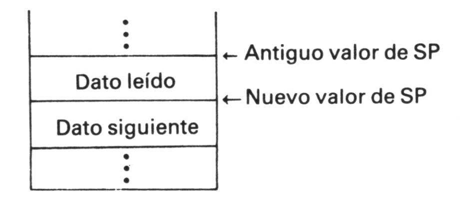
— (SP) ← dato para apilar: colocar el dato en la posición de memoria cuya dirección está en SP. Se dice que SP apunta hacia esta posición de memoria.



A la operación de retirar el dato situado en la cima de la pila se le llama *desapilamiento*. Se resume por las operaciones siguientes:

— Dato leído ← (SP): leer el dato cuya dirección está situada en SP.

— SP ← SP + 2: incrementar en 2 el stack pointer.



La pila puede utilizarse para salvaguardar temporalmente algunos datos en la memoria central. Para ello se apilan sucesivamente todos los datos que se desea resguardar:

Apilar dato 1
Apilar dato 2
Apilar dato 3

Cuando llega el momento de recuperar estos datos, se efectúan tantos desapilamientos como apilamientos fueron realizados, pero en sentido inverso, pues debe tenerse en cuenta la regla: «último en llegar, primero en salir».

Desapilar dato 3
Desapilar dato 2
Desapilar dato 1

El stack también sirve para las llamadas y retornos de los subprogramas. En el momento de una llamada de subprograma, la dirección de retorno se apila, es decir, la dirección que sigue a la llamada del subprograma. En el momento de un retorno de subprograma se desapila la dirección de retorno y se bifurca a esta dirección.

Gracias a esta estructura de stack pueden efectuarse llamadas a subprogramas imbricados (intercalados). La orden de retorno terminará el programa más interior.

En la utilización simultánea del stack por los subprogramas, y para la salvaguarda temporal de datos, habrá que asegurarse de que en el subprograma haya tantos apilamientos de datos como desapilamientos. En caso contrario, el dato presente en la cima de la pila (stack) en el momento del retorno del subprograma no será la dirección de retorno anteriormente apilada, lo que creará el riesgo de un bloqueo de programa.

El último registro del microprocesador Z 8Ø es el contador ordinal (PC: abreviación de Program Counter) que ya hemos visto.

1.5. Modos de direccionado

Los modos de direccionado describen cómo deben tomarse los datos utilizados por una instrucción. Para mejor comprender este concepto tomaremos como ejemplo la instrucción de transferencia. Esta instrucción ha recibido el mnemotécnico LD (abreviación de LOAD: carga en inglés). La instrucción completa se escribe:

```
LD x, y
```

Su efecto es transferir el valor de y a x. Es el LET x = y del Basic, donde los operandos (x e y) son enteros de 8 o 16 bits, y es el operando *fuente*, x es el operando *destino*.

1.5.1. Direccionado por registro

En este tipo de direccionado, el operando es un registro de 8 o 16 bits.

Ejemplo:

78	LD	A, B	transfiere el contenido del
código	mnemónico	operandos	registro B al registro A

En este ejemplo los operandos fuente y destino son, los dos, registros de 8 bits.

1.5.2. Direccionado inmediato

El operando es una constante de 8 o 16 bits que se memoriza en 1 o 2 octetos de la instrucción.

Ejemplos:

```
3E Ø5    LD A,5
```

Esta instrucción transfiere el valor 5 al registro A. El valor 5 es memorizado en el segundo octeto de la instrucción. El operando fuente se obtiene por direccionado inmediato, mientras que el operando destino se obtiene por direccionado de registro. Este último no puede ser del tipo direccionado inmediato ya que debe ser modificado.

```
Ø1 3A 1E    LD BC,1E3AH
```

Esta instrucción transfiere el valor hexadecimal 1E3A (la H en la instrucción indica un valor hexadecimal), al registro doble BC. El dato de 16 bits 1E3A se guarda en el segundo y tercer octeto de la instrucción. El orden de almacenaje en memoria de los dos octetos que contengan el dato es un poco desanimador para un principiante. El segundo octeto 3 A (*el octeto menos significativo* u *octeto de menos peso*) se almacena el primero y en la dirección N + 1. El primer octeto 1E (*el octeto más significativo* o *de más peso*) se almacena en segundo lugar en la dirección N + 2. El código de la instrucción (Ø1) se almacena en la dirección N.

1.5.3. Direccionado directo

En esta forma de direccionado se suministra la dirección de memoria donde se encuentra el operando.

Ejemplos:

```
3A ED 59     LD A,(59EDH)
```

Esta instrucción transfiere el dato de 8 bits situado en la dirección 59EDH al registro A. La dirección es memorizada en los octetos 2 y 3 de la instrucción. Los paréntesis simbolizan el contenido de la dirección.

```
2A ED 59     LD HL,(59EDH)
```

Esta otra instrucción transfiere el dato de 16 bits situado en la dirección 59EDH, al registro doble HL. El octeto situado en la dirección 59EDH se coloca en L y el situado en la dirección 59EEH (dirección siguiente) se coloca en H. De aquí sale la regla de memorización de los números de 16 bits.

```
22 ED 59     LD (59EDH), HL
```

Esta última instrucción realiza la transferencia inversa de la precedente.

1.5.4. Direccionado indirecto por registro

Para esta forma se suministra el registro doble que contiene la dirección del operando de 8 bits.

Ejemplo:

```
7E     LD A,(HL)
```

El dato situado en la dirección que está contenida en HL es transferido a A. Se dice que HL *apunta* hacia este dato. HL es calificado de *indicador* o *puntero.*

Los dos grupos de instrucciones siguientes transfieren el mismo valor a A:

```
LD HL,59EDH                     LD A,(59EDH)
LD A,(HL)
```

1.5.5. Direccionado por índice

Este método es una extensión del precedente. La dirección del operando es igual al contenido de uno de los registros del índice (IX o IY) más un valor de 8 bits que se le proporciona.

Ejemplo:

```
DD 46 Ø9     LD B,(IX + 9)
```

Esta instrucción calcula la dirección del operando añadiéndole 9 al valor contenido en IX. El octeto situado en esta dirección se coloca en el registro B. El valor del desplazamiento es almacenado en el tercer octeto de la instrucción. Este desplazamiento puede ser positivo o negativo. Si el valor del octeto que lo representa es superior a 7FH(127), el desplazamiento será negativo e igual en valor absoluto a 256 menos el valor del octeto. Así, el octeto de valor F7 representa un desplazamiento de −9. Entonces la instrucción se escribe:

```
DD 46 F7     LD B,(IX - 9)
```

Este procedimiento de codificación de números negativos es conocido como codificación *en complemento a 2*. En esta codificación, un número negativo se caracteriza por la puesta a uno del bit más significativo (bit 7 para un número de 8 bits).

1.5.6. Direccionado relativo

Esta clase de direccionado se utiliza con la instrucción de salto relativo JR que provoca un desplazamiento positivo o negativo del contador ordinal.

Ejemplo:

```
18 Ø3     JR $ + 5
```

Esta instrucción provoca un salto de 5 octetos hacia adelante. Se parece a la instrucción GOTO del Basic, excepto en que el punto de bifurcación se da relativo a la posición corriente, y no de forma absoluta como en el Basic. La expresión $ + 5 indica la dirección de la bifurcación ($ es el valor del contador ordinal al principio de la instrucción). El desplazamiento se almacena en el segundo octeto de la instrucción.

Utilizando la codificación en complemento a dos puede especificarse un desplazamiento negativo.

Ejemplo:

```
18 F9     JR $ - 5
```

1.5.7. Direccionado por bit

Algunas instrucciones permiten manipular directamente los bits de un número de 8 bits. SET pone un bit a uno, RES coloca un bit a cero y BIT comprueba el valor de un bit, y posiciona el indicador Z en consecuencia. Los bits son representados por un número del Ø al 7. El bit Ø es el menos significativo y el bit 7 es el más significativo.

Ejemplos:

CB DF	SET 3, A	puesta a 1 del bit 3 de A
CB AE	RES 5, (HL)	puesta a Ø del bit 5 de (HL)
DD CB Ø3 46	BIT Ø, (IX + 3)	comprueba el bit Ø de (IX + 3)

1.6. Juego de instrucciones del Z 8Ø

En este apartado estudiaremos todas las instrucciones disponibles en el Z 8Ø. El número de instrucciones es relativamente grande para un microprocesador de 8 bits (696 instrucciones). Este es uno de los puntos fuertes del Z 8Ø.

Para cada instrucción daremos el detalle de la operación efectuada, el detalle de la eventual modificación de los indicadores del registro F y el número de ciclos de reloj utilizados para realizar la instrucción. Esta cantidad de ciclos permite calcular explícitamente la duración de una instrucción. Sabiendo que la frecuencia de reloj es de 3,25 MHz el tiempo de ciclo es igual a 0,3 µs. La duración de una instrucción es igual al producto del número de ciclos por la duración del ciclo del reloj. Una instrucción simple necesita cuatro ciclos y dura como consecuencia 1,2 µs, o sea un poco más de una millonésima de segundo (el microsegundo, de símbolo µs, es la millonésima parte de un segundo). Percibimos así la extrema rapidez del lenguaje máquina en comparación con el Basic.

1.6.1. Transferencia de 8 bits

La instrucción LD d,s que ya hemos visto, realiza la transferencia del operando s al operando d. Estos operandos se describen en uno de los modos de direccionado del Z 8Ø. No obstante, todas las combinaciones de modos de direccionado no están permitidas para s y d. Examinemos la lista de instrucciones de este tipo que están permitidas:

Mnemotécnicos	*Operación*	*Código de la instrucción binaria*	*Hexadecimal*	*Número de ciclos de reloj*	*Indicadores*
LD r, s	r ← s	Ø 1 ← r → ← s →		4	no se modifican
LD r, n	r ← n	Ø Ø ← r → 1 1 Ø ← n →		7	no se modifican
LD r, (HL)	r ← (HL)	Ø 1 ← r → 1 1 Ø		7	no se modifican
LD r, (IX+d)	r ← (IX+d)	1 1 Ø 1 1 1 Ø 1 Ø 1 ← r → 1 1 Ø ← d →	DD	19	no se modifican
LD r, (IY+d)	r ← (IY+d)	1 1 1 1 1 1 Ø 1 Ø 1 ← r → 1 1 Ø ← d →	FD	19	no se modifican
LD (HL), r	(HL) ← r	Ø 1 1 1 Ø ← r →		7	no se modifican
LD (IX+d), r	(IX+d) ← r	1 1 Ø 1 1 1 Ø 1 Ø 1 1 1 Ø ← r → ← d →	DD	19	no se modifican
LD(IY+d), r	(IY+d) ← r	1 1 1 1 1 1 Ø 1 Ø 1 1 1 Ø ← r → ← d →	FD	19	no se modifican
LD (HL), n	(HL) ← n	Ø Ø 1 1 Ø 1 1 Ø ← n →	36	1Ø	no se modifican
LD (IX+d), n	(IX+d) ← n	1 1 Ø 1 1 1 Ø 1 Ø Ø 1 1 Ø 1 1 Ø ← d → ← n →	DD 36	19	no se modifican
LD (IY+d), n	(IY+d) ← n	1 1 1 1 1 1 Ø 1 Ø Ø 1 1 Ø 1 1 Ø ← d → ← n →	FD 36	19	no se modifican
LD A, (BC)	A ← (BC)	Ø Ø Ø Ø 1 Ø 1 Ø	ØA	7	no se modifican
LD A, (DE)	A ← (DE)	Ø Ø Ø 1 1 Ø 1 Ø	1A	7	no se modifican

LD A, (nn)	A ← (nn)	Ø Ø 1 1 1 Ø 1 Ø ← n: menos peso → ← n: más peso →	3A	13	no se modifican
LD (BC), A	(BC) ← A	Ø Ø Ø Ø Ø Ø 1 Ø	Ø2	7	no se modifican
LD (DE), A	(DE) ← A	Ø Ø Ø 1 Ø Ø 1 Ø	12	7	no se modifican
LD (nn), A	(nn) ← A	Ø Ø 1 1 Ø Ø 1 Ø ← n: menos peso → ← n: más peso →	32	13	no se modifican
LD I, A	I ← A	1 1 1 Ø 1 1 Ø 1 Ø 1 Ø Ø Ø 1 1 1	ED 47	9	no se modifican
LD R, A	R ← A	1 1 1 Ø 1 1 Ø 1 Ø 1 Ø Ø 1 1 1 1	ED 4F	9	no se modifican
LD A, I	A ← I	1 1 1 Ø 1 1 Ø 1 Ø 1 Ø 1 Ø 1 1 1	ED 57	9	S y Z posicionados según el estado de A; H y N puestos a Ø; C sin afectar; P/V contienen el estado de biestable de interrupciones.
LD A, R	A ← R	1 1 1 Ø 1 1 Ø 1 Ø 1 Ø 1 1 1 1 1	ED 5F	9	Indicadores modificados como en el caso precedente.

En este cuadro *n* designa una constante de 8 bits, *d* un desplazamiento de 8 bits positivo o negativo (direccionado por índice), *nn* una constante de 16 bits y *r* o *s* un registro de 8 bits. Estos registros son codificados por el Z 8Ø sobre tres bits que también se encuentran en el código de la instrucción. La correspondencia entre los registros y estos tres bits se resume en el cuadro siguiente.

Registro	*r o s*
B	ØØØ
C	ØØ1
D	Ø1Ø
E	Ø11
H	1ØØ
L	1Ø1
A	111

Una lista exhaustiva de estas instrucciones se encuentra en el anexo.

Estas instrucciones de transferencia, que son fáciles de comprender, de hecho son las más empleadas en un programa. Según la forma de direccionado utilizada por los operandos, el número de octetos de la instrucción, así como la duración de la instrucción, varían. Estas instrucciones, salvo las dos últimas, no afectan a los indicadores.

1.6.2. Transferencia de 16 bits

Examinemos ahora la lista menos importante de las instrucciones de transferencia de 16 bits.

Mnemotécnico	*Operación*	*Código de la instrucción binaria*	*Hexadecimal*	*Número de ciclos de reloj*	*Indicadores*
LD dd, nn	dd ← nn	Ø Ø ←dd→ Ø Ø Ø 1 ← n: menos peso → ← n: más peso →		1Ø	
LD IX, nn	IX ← nn	1 1 Ø 1 1 1 Ø 1 Ø Ø 1 Ø Ø Ø Ø 1 ← n: menos peso → ← n: más peso →	DD 21	14	no se modifican
LD IY,nn	IY ← nn	1 1 1 1 1 1 Ø 1 Ø Ø 1 Ø Ø Ø Ø 1 ← n: menos peso → ← n: más peso →	FD 21	14	no se modifican
LD HL, (nn)	H ← (nn+1) L ← (nn)	Ø Ø 1 Ø 1 Ø 1 Ø ← n: menos peso → ← n: más peso →	2A	16	no se modifican
LD dd, (nn)	dd ← (nn)	1 1 1 Ø 1 1 Ø 1 Ø 1 ←dd→ 1 Ø 1 1 ← n: menos peso → ← n: más peso →	ED	2Ø	no se modifican
LD IX, (nn)	IX ← (nn)	1 1 Ø 1 1 1 Ø 1 Ø Ø 1 Ø 1 Ø 1 Ø ← n: menos peso → ← n: más peso →	DD 2A	2Ø	no se modifican

LD IY, (nn)	IY ← (nn)	1 1 1 1 1 1 Ø 1 Ø Ø 1 Ø 1 Ø 1 Ø ← n: menos peso → ← n: más peso →	FD 2A	2Ø	no se modifican
LD (nn), HL	(nn+1) ← H (nn) ← L	Ø Ø 1 Ø Ø Ø 1 Ø ← n: menos peso → ← n: más peso →	22	16	no se modifican
LD (nn), dd	(nn) ← dd	1 1 1 Ø 1 1 Ø 1 Ø 1 ←d d→ Ø Ø 1 1 ← n: menos peso → ← n: más peso →	ED	2Ø	no se modifican
LD (nn), IX	(nn) ← IX	1 1 Ø 1 1 1 Ø 1 Ø Ø 1 Ø Ø Ø 1 Ø ← n: menos peso → ← n: más peso →	DD 22	2Ø	no se modifican
LD (nn), IY	(nn) ← IY	1 1 1 1 1 1 Ø 1 Ø Ø 1 Ø Ø Ø 1 Ø ← n: menos peso → ← n: más peso →	FD 22	2Ø	no se modifican
LD SP, HL	SP ← HL	1 1 1 1 1 Ø Ø 1	F9	6	no se modifican
LD SP, IX	SP ← IX	1 1 Ø 1 1 1 Ø 1 1 1 1 1 1 Ø Ø 1	DD F9	1Ø	no se modifican
LD SP, IY	SP ← IY	1 1 1 1 1 1 Ø 1 1 1 1 1 1 Ø Ø 1	FD F9	1Ø	no se modifican

En este cuadro *nn* designa un número de 16 bits y *dd* uno de los registros de 16 bits: BC, DE, HL, SP. Estos registros se codifican sobre dos bits que se encuentran en el código de la instrucción.

Registro	*dd*
BC	ØØ
DE	Ø1
HL	1Ø
SP	11

Los modos de direccionado posibles, con las instrucciones de transferencia de 16 bits, son menos importantes que los disponibles en las instrucciones de transferencia de 8 bits. No encontramos más que el modo directo y el modo por registro.

Estas instrucciones son muy útiles para manipular los datos de 16 bits o las direcciones de memoria codificadas en 16 bits. Se las emplea en particular para iniciar los registros dobles en vistas a utilizar el modo de direccionado indirecto sobre los números de 8 bits.

1.6.3. Operaciones aritméticas de 8 bits

El Z 8Ø es capaz de efectuar operaciones aritméticas simples sobre números de 8 bits, codificados o no, en complemento a 2. Este tipo de codificación, que ya hemos visto, permite representar en un octeto todos los números enteros comprendidos entre −128 y +127. Los números positivos se codifican normalmente mientras que los números negativos son representados por un número igual a 256 menos su valor. Así, los números de −128 a −1 se representan por los números de 128 a 255 (representación normal).

Valor del octeto en representación normal	Ø	1	...	126	127	128	129	...	254	255
Valor del octeto en complemento a dos	Ø	1	...	126	127	−128	−127	...	−2	−1

Con esta codificación, un número negativo se reconoce por la presencia del valor 1 en el bit más significativo del octeto. Este es el bit que comprueba el indicador de signo S en el momento de las operaciones aritméticas.

De hecho, el microprocesador no hace diferencia entre estos dos tipos de codificaciones. Todas las operaciones aritméticas se realizan sobre la representación normal de números, con pérdida del acarreo o sin ella, en caso de desbordamiento. Esta posibilidad de acarreo asegura la equivalencia de los cálculos para los dos tipos de representación.

Ejemplos: Tomemos por caso la suma de 12 con −4. −4 es codificado por 256 − 4 = 252 en representación normal. La suma sería en binario:

	Representación normal	*Representación en complemento a 2*
∅ ∅ ∅ ∅ 1 1 ∅ ∅	12	12
+ 1 1 1 1 1 1 ∅ ∅	252	−4
(1) ∅ ∅ ∅ ∅ 1 ∅ ∅ ∅	264	8

C ← (1) Los 8 bits resultantes puestos en A

El número resultante contiene 9 bits. Sólo los 8 bits menos significativos se conservan. El último bit se coloca en el indicador de acarreo C. El número resultante es pues igual a 8 si no se tiene en cuenta el acarreo (cálculo en complemento a 2) y a 256 + 8 = 264, si se tiene en cuenta (cálculo en representación normal: 12 + 252 = 264).

Corresponde al usuario el tener o no tener en cuenta la indicación del acarreo, según el tipo de datos que manipule.

Tomemos como otro ejemplo la suma de − 1 y de − 2 que vale − 3. Esta operación sería en representación normal:

$$256 - 1 + 256 - 2 = \underbrace{256}_{\text{acarreo}} + \underbrace{256 - 3}_{\text{valor sobre 8 bits}}$$

El resultado de 8 bits vale 256 − 3, o sea el valor − 3 en complemento a dos, y el indicador C indica un acarreo.

Si el usuario trabaja en representación normal deberá sumar los números 255 y 254 y obtendrá el resultado de 256 (acarreo a 1) más 253, o sea 5∅9 que es la suma de 255 y de 254.

En la práctica se utilizará la codificación en complemento a dos para números comprendidos entre − 128 y + 127 y la codificación en representación normal para números positivos pudiendo sobrepasar el valor 127.

El calificativo de complemento a dos de la codificación proviene del hecho de que la representación normal de un número negativo puede obtenerse haciendo el *complemento* de cada uno de los bits del número positivo correspondiente y añadiendo uno al resultado. La operación de complemento consiste en cambiar los ∅ por 1 y los 1 por ∅.

Ejemplo:

3 se escribe en binario	: ∅ ∅ ∅ ∅ ∅ ∅ 1 1
El complemento de 3 vale	: 1 1 1 1 1 1 ∅ ∅
Añadamos 1 al resultado	: 1 1 1 1 1 1 ∅ 1

El resultado obtenido es 256 − 3 en representación de complemento a 2.

El cuadro siguiente describe las instrucciones de suma y resta sobre 8 bits.

Mnemotécnico	*Operación*	*Código interno*	*Indicadores*
ADD A, s	A ← A + s	Ø Ø Ø	S puesto a 1 si hay resultado negativo, si no puesto a Ø. Z puesto a 1 si hay resultado nulo, si no puesto a Ø. H puesto a 1 si hay acarreo del bit 3, si no a Ø. P/V puesto a 1 si hay acarreo del bit 6, si no puesto a Ø. N puesto a Ø para una suma y a 1 para una resta. C puesto a 1 si hay acarreo del bit 7, si no puesto a Ø.
ADC A, s	A ← A + s + Carry	Ø Ø 1	
SUB s	A ← A − s	Ø 1 Ø	
SBC A, s	A ← A − s − Carry	Ø 1 1	

Todas estas operaciones aritméticas se realizan entre el registro A y el dato de 8 bits s; el resultado se coloca en A con posicionamiento de los indicadores según el resultado.

El símbolo s designa uno de los cinco modos de direccionado indicados en el siguiente cuadro; el código completo de la instrucción se obtiene por inserción del código interno en uno de los códigos siguientes:

Código de la instrucción	*s*	*Modo de direccionado*	*Número de ciclos*
Registro	r	1 Ø [código interno] ← r →	4
Dato de 8 bits	n	1 1 [código interno] 1 1 Ø ← n →	7
Indirecto HL	(HL)	1 Ø [código interno] 1 1 Ø	7
Por índice IX	(IX + d)	1 1 Ø 1 1 1 Ø 1 DD 1 Ø [código interno] 1 1 Ø ← d →	19
Por índice IY	(IY + d)	1 1 1 1 1 1 Ø 1 FD 1 Ø [código interno] 1 1 Ø ← d →	19

El indicador P/V permite saber si ha habido un acarreo del bit 6 de A; dicho en otros términos, si ha habido un desbordamiento en los cálculos para números codificados en complemento a dos:

Ejemplo: 1ØØ + 84 = $\underbrace{128}_{\text{bit de desbordamiento}} + \underbrace{56}_{\text{resto sobre 7 bits}}$

	Binario	Decimal
	Ø 1 1 Ø Ø 1 Ø Ø	1ØØ
+	Ø 1 Ø 1 Ø 1 Ø Ø	84
	1 ⋮ Ø 1 1 1 Ø Ø Ø	184

desbordamiento (1) — resultado sobre 7 bits (Ø 1 1 1 Ø Ø Ø)

El indicador P/V permite saber que en este caso el resultado no es − 72 (184 en representación normal) sino 128 + 56.

Por lo tanto, si se trabaja con números codificados en complemento a dos, es necesario fijarse en el indicador P/V, y no en el indicador Carry utilizado en la representación normal.

Otras dos instrucciones utilizadas frecuentemente permiten el incremento o decremento del operador en una unidad.

Mnemotécnico	*Operación*	*Indicadores*
INC s	s ← s + 1	Los indicadores S, Z, H y P/V son modificados de la misma forma que en las otras instrucciones aritméticas. N es puesto a 1 para la instrucción INC y a Ø para DEC. C no se modifica.
DEC s	s ← s − 1	

El cuadro siguiente resume los modos de direccionado posibles:

Modo de direccionado	*s*	*Código de la instrucción*	*Número de ciclos*
Registro	r	Ø Ø ← r → 1 Ø X	4
Indirecto HL	(HL)	Ø Ø 1 1 Ø 1 Ø X	11
Por índice IX	(IX + d)	1 1 Ø 1 1 1 Ø 1 DD Ø Ø 1 1 Ø 1 Ø X ← d →	23
Por índice IY	(IY + d)	1 1 1 1 1 1 Ø 1 FD Ø Ø 1 1 Ø 1 Ø X ← d →	23

El bit X vale ∅ para la instrucción INC y 1 para la instrucción DEC.

Finalmente, la instrucción CP permite comparar el valor del acumulador con el operando. Esta instrucción efectúa la sustracción A − s y proporciona los indicadores según el resultado. Este último no se copia en el registro A que no sufre modificación.

Mnemotécnico	*Operación*	*Código interno*
CP s	test A − s	1 1 1

Los modos de direccionado son los mismos que para las operaciones aritméticas ADD y SUB.

Ejemplo:

CP 3∅

Esta instrucción compara el acumulador con el valor 3∅.

$Z = 1$ si $A = 3\emptyset$
$C = 1$ si $A < 3\emptyset$
$C = \emptyset$ si $A \geqslant 3\emptyset$

Utilizando las informaciones suministradas por los indicadores Z y C, un valor de 8 bits contenido en el registro A puede compararse con el valor de 8 bits representado por el operando.

1.6.4. Operaciones lógicas de 8 bits

Las instrucciones lógicas efectúan una operación lógica entre cada uno de los bits del acumulador y cada uno de los bits del operando. Es posible efectuar tres operaciones lógicas diferentes entre dos bits:

El operador AND hace el «Y lógico» entre los dos bits operandos. El resultado solamente es 1 si el primero *y* el segundo operando están a 1.

El operador OR hace la «O lógica» entre los dos operandos. El resultado es 1 si el primero *o* el segundo operando están a 1.

El operador XOR hace la «O exclusiva» entre los dos operandos. El resultado será 1 si el primero *o* el segundo operando están a 1, pero no los dos al mismo tiempo.

La tabla de la verdad siguiente resume estas funciones:

Primer operando	*Segundo operando*	AND	*OR*	*XOR*
Ø	Ø	Ø	Ø	Ø
Ø	1	Ø	1	1
1	Ø	Ø	1	1
1	1	1	1	Ø

Las instrucciones lógicas correspondientes del Z 8Ø efectúan estas operaciones en paralelo sobre los 8 bits del acumulador y del operando, y dejan el resultado en el acumulador.

Mnemotécnico	*Operación*	*Código interno*	*Indicadores*
AND s	A ← A Y s	1 Ø Ø	S puesto a 1 si hay resultado negativo. Z puesto a 1 si hay resultado nulo.
OR s	A ← A O s	1 1 Ø	H puesto a 1 para AND y si no a Ø. P/V contiene la paridad del resultado.
XOR s	A ← A O s exclusiva	1 Ø 1	N puesto a cero. C puesto a cero.

Los mandos de direccionado son los mismos que para las instrucciones ADD y SUB.

Estas instrucciones son de mucha utilidad para acceder a cierto número de bits del registro A.

La instrucción AND sirve para aislar cierto número de bits del registro A, colocando los otros bits a cero. Se realiza así una máscara para seleccionar estos bits.

Ejemplo:

AND 15 permite aislar los cuatro bits de menor peso de A.

Si A contiene un 39H, contendrá un 9 después de la ejecución de esta instrucción. Solamente los 4 bits de menor peso han sido guardados; los otros han sido puestos a cero.

La instrucción OR provoca la puesta a 1 de unos cuantos bits del acumulador.

Ejemplo:

OR FØH provoca la puesta a 1 de los 4 bits de más peso de A.

Si A contiene un 39H, contendrá un F9H después de la ejecución de esta instrucción.

La instrucción OR A permite probar si el acumulador es nulo sin modificar este último, lo cual será empleado a menudo. También provoca la puesta a cero del indicador Carry, lo que será útil para el empleo de ADC y SBC.

La instrucción XOR sirve para complementar cierto número de bits de A.

Ejemplo:

XOR FØH complementa los 4 bits de más peso de A.

Si en A hubiera un 39H, contendría un C9H después de la ejecución de esta instrucción.

La instrucción XOR A provoca en particular la puesta a cero de todos los bits de A, o sea la puesta a cero de A. Esto equivale a LD A,Ø, pero presenta la ventaja de no ocupar más que un octeto en lugar de dos y de posicionar los indicadores.

1.6.5. Decalados

El Z 8Ø es capaz de decalar hacia la derecha o hacia la izquierda un número de 8 bits. Esta operación consiste en desplazar cada uno de los bits del número hacia la derecha o hacia la izquierda. El bit sobrante del decalado se utiliza de diferentes maneras según la instrucción aplicada.

Ejemplos:

1 Ø Ø 1 1 1 1 Ø (9E) decalado a la izquierda: Ø Ø 1 1 1 1 Ø Ø (3C)

1 Ø Ø 1 1 1 1 Ø (9E) decalado a la derecha : Ø 1 Ø Ø 1 1 1 1 (4F)

El bit saliente es el bit 7 después del decalado a la izquierda y el bit Ø después del decalado a la derecha. Este bit puede colocarse en el Carry o reinsertarse en el bit opuesto del número para obtener un decalado circular.

Sin la pérdida del bit saliente, el decalado a la izquierda es equivalente a una multiplicación por dos y el decalado a la derecha a una división por dos. En efecto, esta operación decala las cifras de un número escrito en base dos, lo que se convierte en una multiplicación o división del número por la base, que vale 2. Utilizaremos esta propiedad en la realización de subprogramas de multiplicación y de división.

Ahora examinemos el detalle de las instrucciones de decalado:

Mnemotécnico	*Operación*	*Código*		*Número de ciclos*	*Indicadores*
RLCA	C ← 7 ← Ø (A)	Ø Ø Ø Ø Ø 1 1 1	Ø7	4	S, Z y V sin modificar. H y N puesto a cero. C contiene el bit saliente.
RLA	C ← 7 ← Ø (A)	Ø Ø Ø 1 Ø 1 1 1	17	4	
RRC A	7 → Ø (A) → C	Ø Ø Ø Ø 1 1 1 1	ØF	4	
RRA	7 → Ø (A) → C	Ø Ø Ø 1 1 1 1 1	1F	4	
RLD	7-4 3-Ø (A) 7-4 3-Ø (HL)	1 1 1 Ø 1 1 Ø 1 Ø 1 1 Ø 1 1 1 1	ED 6F	18	S y Z posicionados según el valor resultante de A. H y N puestos a cero. C sin modificar. P/V contiene la paridad de A.
RRD	7-4 3-Ø (A) 7-4 3-Ø (HL)	1 1 1 Ø 1 1 Ø 1 Ø 1 1 Ø Ø 1 1 1	ED 67	18	

Las cuatro primeras instrucciones de este cuadro efectúan rotaciones en el acumulador. Las dos últimas son más especiales. Actúan sobre grupos de 4 bits contenidos en el registro A y en el octeto situado en la dirección HL. Sirven especialmente para la manipulación de los números codificados en *BCD (código decimal binario)* utilizados en aritmética decimal. Las cifras decimales son así codificadas sobre 4 bits. Así pues, un número decimal de dos cifras puede colocarse en un octeto.

Ejemplo:

72 decimal se escribe en BCD: Ø 1 1 1 Ø Ø 1 Ø o sea 72H
cifra 7 cifra 2

Las instrucciones RRD y RLD permiten realizar decalados de números decimales.

Las otras instrucciones de rotación con 8 bits se resumen en el siguiente cuadro:

Mnemotécnico	*Operación*	*Código interno*	*Indicadores*
RLC s	C ← [7 ← Ø] ← (s)	Ø Ø Ø	Z puesto a 1 si el resultado es nulo.
RL s	→ C ← [7 ← Ø] ← (s)	Ø 1 Ø	S puesto a 1 si el resultado es negativo.
RRC s	→ [7 → Ø] → C (s)	Ø Ø 1	H puesto a Ø.
RR s	→ [7 → Ø] → C → (s)	Ø 1 1	P/V contiene la paridad del resultado.
SLA s	C ← [7 ← Ø] ← Ø (s)	1 Ø Ø	N puesto a Ø.
SRA s	[7 → Ø] → C (s)	1 Ø 1	C contiene el bit saliente del decalado.
SRL s	Ø → [7 → Ø] → C (s)	1 1 1	

El signo s designa uno de los cuatro modos de direccionado en el cuadro siguiente; el código completo de la instrucción se obtiene entonces por inserción del código interno en uno de los códigos siguientes:

Modo de direccionado	*s*	*Código*	*Números de ciclos*
Registro	r	1 1 Ø Ø 1 Ø 1 1 CB Ø Ø [código interno] ← r →	8
Indirecto HL	(HL)	1 1 Ø Ø 1 Ø 1 1 CB Ø Ø [código interno] 1 1 Ø	15
Por índice IX	(IX + d)	1 1 Ø 1 1 1 Ø 1 DD 1 1 Ø Ø 1 Ø 1 1 CB ← d → Ø Ø [código interno] 1 1 Ø	23
Por índice IY	(IY + d)	1 1 1 1 1 1 Ø 1 FD 1 1 Ø Ø 1 Ø 1 1 CB ← d → Ø Ø [código interno] 1 1 Ø	23

La correspondencia entre el nombre del registro r y los 3 bits del código de la instrucción se da en el párrafo 1.6.1.

1.6.6. Instrucciones que actúan sobre un bit

El Z 8Ø posee tres instrucciones que permiten posicionar o comprobar un bit de un número de ocho bits.

Mnemotécnico	*Operación*	*Código interno*	*Indicadores*
BIT b, s	Comprobación del bit b de s.	Ø 1	Z está a 1 si el bit vale Ø. H es puesto a 1, N es puesto a Ø. S y P/V tienen un valor cualquiera. C no es modificado.
SET b, s	Puesta a 1 del bit b de s.	1 1	Los indicadores no son modificados.
RES b, s	Puesta a Ø del bit b de s.	1 Ø	Los indicadores no son modificados.

El número del bit b varía entre Ø y 7. Este número se encuentra en tres de los bits del código de la instrucción, según la correspondencia siguiente:

b	*Código de 3 bits*
Ø	Ø Ø Ø
1	Ø Ø 1
2	Ø 1 Ø
3	Ø 1 1
4	1 Ø Ø
5	1 Ø 1
6	1 1 Ø
7	1 1 1

La tabla siguiente resume los diferentes modos de direccionado posibles para s.

Modo de direccionado	*s*	*Código de la instrucción*		*Número de ciclos*
Registro	r	1 1 Ø Ø 1 Ø 1 1	CB	8
		código interno ← b →← r →		
Indirecto HL	(HL)	1 1 Ø Ø 1 Ø 1 1	CB	12 para BIT
		código interno ← b → 1 1 Ø		15 para SET y RES
Por índice IX	(IX + d)	1 1 Ø 1 1 1 Ø 1	DD	2Ø para BIT
		1 1 Ø Ø 1 Ø 1 1	CB	23 para SET y RES
		← d →		
		código interno ← b → 1 1 Ø		
Por índice IY	(IY + d)	1 1 1 1 1 1 Ø 1	FD	2Ø para BIT
		1 1 Ø Ø 1 Ø 1 1	CB	23 para SET y RES
		← d →		
		código interno ← b → 1 1 Ø		

1.6.7. Instrucciones de uso general

Mnemotécnico	*Operación*	*Código de la instrucción*	*Número de ciclos*	*Indicadores*
NOP	No hay operación	Ø Ø Ø Ø Ø Ø Ø Ø ØØ	4	No se modifican.
SCF	Carry ← 1	Ø Ø 1 1 Ø 1 1 1 37	4	C a 1, H y N a Ø. Otros indicadores no se modifican.
CCF	Complemento del Carry	Ø Ø 1 1 1 1 1 1 3F	4	H desconocido. C complementado, N a Ø. Otros indicadores no se modifican.
CPL	Complemento de A $A \leftarrow \overline{A}$	Ø Ø 1 Ø 1 1 1 1 2F	4	H y N a 1. Otros indicadores no se modifican.
NEG	Negación de A en complemento a 2 $A \leftarrow \overline{A} + 1$	1 1 1 Ø 1 1 Ø 1 ED Ø 1 Ø Ø Ø 1 Ø Ø 44	8	N puesto a 1, los otros indicadores son posicionados según el resultado de A. P/V contiene el desbordamiento eventual del bit 6.
DAA	Ajuste decimal	Ø Ø 1 Ø Ø 1 1 1 27	4	N no se modifica, P/V contiene la paridad del resultado. Los otros indicadores son posicionados según el resultado de A.
HALT	Paro del Z 8Ø	Ø 1 1 1 Ø 1 1 Ø 76	4	Indicadores sin modificar.

La instrucción NOP será de utilidad durante la puesta a punto. Gracias a ella podrá calcularse el efecto de una secuencia de instrucciones reemplazándolas por octetos nulos (instrucción NOP).

La instrucción DAA efectúa automáticamente el ajuste decimal que debe ser realizado a continuación de una suma o de una resta de dos números decimales codificados en BCD. Esta instrucción utiliza las informaciones facilitadas por los indicadores H y N.

Ejemplo: Consideremos la suma de 34 y 39:

```
LD  A,34H     ; A contiene 34 en BCD
ADD A,29H     ; A contiene 5 DH que no es un número en BCD
DAA           ; A contiene 63 en BCD que es el resultado de la suma
```

Después de una suma, la instrucción comprueba si la cifra de menos peso (4 bits de menor peso) es superior a 9, en cuyo caso se le añade 6 al número para obtener la nueva cifra de las unidades e incrementar la cifra de las decenas (4 bits de más peso). El mismo procedimiento sirve para la cifra de las decenas, pero añadiendo esta vez 6 * 16 al número y posicionando el indicador Carry en caso de desbordamiento.

Esta instrucción, que es muy rápida, facilita considerablemente el empleo de los números decimales codificados en BCD. No obstante, tan sólo funciona con números de 8 bits, lo que hace más delicada la manipulación de números decimales de mayor tamaño.

1.6.8. Instrucciones aritméticas sobre 16 bits

Una de las ventajas del microprocesador Z 8Ø sobre sus predecesores es su capacidad para efectuar sumas y restas sobre números enteros de 16 bits. Las instrucciones se encuentran resumidas en el siguiente cuadro:

Mnemotécnico	*Operación*	*Código de la instrucción*	*Número de ciclos*	*Indicadores*
ADD HL, ss	HL ← HL + ss	Ø Ø ←s s→ 1 Ø Ø 1	11	N puesto a cero, H desconocido. S, Z, P/V no se modifican. C vale 1 si hay un acarreo del bit 15 de HL.
ADC HL, ss	HL ← HL + ss + Carry	1 1 1 Ø 1 1 Ø 1 ED Ø 1 ←s s→ 1 Ø 1 Ø	15	N puesto a Ø, H desconocido. S contiene el bit 15 de HL. Z vale 1 si HL es nulo. P/V vale 1 si hay un acarreo del bit 14. C vale 1 si hay un acarreo del bit 15.
SBC HL, ss	HL ← HL − ss − Carry	1 1 1 Ø 1 1 Ø 1 ED Ø 1 ←s s→ Ø Ø 1 Ø	15	N puesto a 1. Los otros indicadores varían de la misma forma que en la instrucción ADC.

ADD IX, pp	IX ← IX + pp	1 1 Ø 1 1 1 Ø 1	DD	15	N puesto a cero. H desconocido. S, Z, P/V no se modifican. C vale 1 si hay un acarreo del bit 15 de HL.
		Ø Ø ←p p→ 1 Ø Ø 1			
ADD IY, rr	IY ← IY + rr	1 1 1 1 1 1 Ø 1	FD	15	
		Ø Ø ←r r→ 1 Ø Ø 1			
INC ss	ss ← ss + 1	Ø Ø ←s s→ Ø Ø 1 1		6	No se modifican.
INC IX	IX ← IX + 1	1 1 Ø 1 1 1 Ø 1	DD	1Ø	
		Ø Ø 1 Ø Ø Ø 1 1	23		
INC IY	IY ← IY + 1	1 1 1 1 1 1 Ø 1	FD	1Ø	
		Ø Ø 1 Ø Ø Ø 1 1	23		
DEC ss	ss ← ss − 1	Ø Ø ←s s→ 1 Ø 1 1		6	
DEC IX	IX ← IX − 1	1 1 Ø 1 1 1 Ø 1	DD	1Ø	
		Ø Ø 1 Ø 1 Ø 1 1	2B		
DEC IY	IY ← IY − 1	1 1 1 1 1 1 Ø 1	FD	1Ø	
		Ø Ø 1 Ø 1 Ø 1 1	2B		

La notación en complemento a dos de números negativos es igualmente utilizable para números de 16 bits. Así se pueden codificar todos los números enteros comprendidos entre − 32768 y 32767; un número negativo representado por un número de 16 bits vale 65536 más el valor del número negativo. El bit 15 permite saber el signo del número (número negativo si el bit 15 es 1).

En estas operaciones aritméticas los indicadores comprueban el valor del resultado que es un número de 16 bits y ya no está en el registro A. El indicador Carry permitirá saber si hay un acarreo que provenga del bit 15 del resultado.

Los símbolos ss, pp y rr designan registros de 16 bits, resumidos en las tablas de más abajo, las cuales dan la correspondencia entre el nombre del registro y los dos bits que lo representan en el código de la instrucción.

Registro	*ss*
BC	ØØ
DE	Ø1
HL	1Ø
SP	11

Registro	*pp*
BC	ØØ
DE	Ø1
IX	1Ø
SP	11

Registro	*rr*
BC	ØØ
DE	Ø1
IY	1Ø
SP	11

1.6.9. Instrucciones de apilamiento (stack) y de desapilamiento

El Z 8Ø posee instrucciones para apilar o desapilar registros de 16 bits, siendo el registro SP el puntero de pila (stack).

La operación de apilamiento se resume por las acciones siguientes:

SP ← SP − 1	:SP es decrementado en 1.
(SP) ← octeto de más peso	:El octeto de más peso del registro es :apilado (colocado en la dirección :contenida en SP).
SP ← SP − 1	:SP es decrementado en 1.
(SP) ← octeto de menor peso	:El octeto de menor peso del registro es :apilado (colocado en la dirección :contenida en SP).

La operación inversa de desapilamiento se resume:

Octeto de menos peso ← (SP)	:El octeto de menos peso del registro es :desapilado.
SP ← SP + 1	:SP es incrementado en 1.
Octeto de más peso ← (SP)	:El octeto de más peso es desapilado.
SP ← SP + 1	:SP es incrementado en 1.

El cuadro siguiente da las instrucciones de apilamiento (PUSH) y de desapilamiento (POP) del Z 8Ø.

Mnemotécnico	*Operación*	*Código de la instrucción*	*Número de ciclos*	*Indicadores*
PUSH qq	Apilamiento de qq	1 1 ←q q→ Ø 1 Ø 1	11	No se modifican
PUSH IX	Apilamiento de IX	1 1 Ø 1 1 1 Ø 1 DD 1 1 1 Ø Ø 1 Ø 1 E5	15	
PUSH IY	Apilamiento de IY	1 1 1 1 1 1 Ø 1 FD 1 1 1 Ø Ø 1 Ø 1 E5	15	
POP qq	Desapilamiento de qq	1 1 ←q q→ Ø Ø Ø 1	1Ø	
POP IX	Desapilamiento de IX	1 1 Ø 1 1 1 Ø 1 DD 1 1 1 Ø Ø Ø Ø 1 E1	14	
POP IY	Desapilamiento de IY	1 1 1 1 1 1 Ø 1 FD 1 1 1 Ø Ø Ø Ø 1 E1	14	

Los valores posibles del registro doble qq son los siguientes:

Registro	*qq*
BC	ØØ
DE	Ø1
HL	1Ø
AF	11

Estas instrucciones son útiles para guardar temporalmente el valor de un registro de 16 bits (PUSH) y restaurar este valor (POP) cuando sea necesario.

Cuando se utilicen estas instrucciones será necesario asegurarse de que el stack pointer contiene la dirección de una zona de memoria no utilizada. Será interesante poner una instrucción de inicialización del registro SP (LD SP, dirección zona libre) en cabeza de su programa.

Dado que los apilamientos se hacen decrementando el puntero del stack, el registro SP deberá ser iniciado a la dirección más alta de la zona libre (fondo de la pila).

1.6.10. Instrucciones de bifurcación

El Z 8Ø posee instrucciones de bifurcación incondicional que permiten saltar a una dirección de memoria sea cual fuere el estado de los indicadores (instrucciones análogas al GOTO del Basic). Igualmente tiene instrucciones de bifurcación condicional que sólo efectúan el salto si uno de los indicadores se encuentra en el estado deseado (es el equivalente al IF indicador = estado deseado THEN GOTO). Estas últimas instrucciones por lo general se colocan detrás de una instrucción de comprobación como CP.

Ejemplo:

```
CP 2          ;A ¿es igual a 2?
JP Z,3E8H     ;si es sí saltar a 3E8H, si no pasar a la instrucción
              ;siguiente.
```

La dirección de bifurcación puede ser definida de manera absoluta colocándola en dos octetos de la instrucción de salto (instrucción JP), o bien, de manera relativa, dándole el desplazamiento comprendido entre − 128 y + 127 que separa la dirección de bifurcación de la

dirección actual (instrucción JR) o también de manera indirecta mediante los registros HL, IX o IY (instrucción JP).

En la práctica, siempre que sea posible, tendremos interés en utilizar una instrucción de bifurcación relativa que sólo ocupa dos octetos, mejor que no una instrucción de bifurcación absoluta que ocupa tres octetos. Esta última se utilizará si la dirección de bifurcación se encuentra a más de 128 octetos de la dirección actual.

Mnemotécnico	*Operación*	*Código de la instrucción*	*Número de ciclos*	*Indicadores*
JP nn	Bifurcación absoluta a la dirección nn. PC ← nn	1 1 Ø Ø Ø Ø 1 1 C3 ←n: menos peso→ ←n: más peso →	1Ø	No se modifican
JP cc, nn	Bifurcación condicional absoluta a la dirección nn. Si condición cc: PC ← nn	1 1 ←cc→ Ø 1 Ø ←n: menos peso→ ←n: más peso →	1Ø	
JR d	Bifurcación relativa de d octetos a partir de la instrucción siguiente a JR. PC ← PC + d	Ø Ø Ø 1 1 Ø Ø Ø 18 ←— d —→	12	
JR C, d	Si C = 1 PC ← PC + d Si no continuar	Ø Ø 1 1 1 Ø Ø Ø 38 ←— d —→	12 si C = 1 7 si C = Ø	
JR NC, d	Si C = Ø PC ← PC + d Si no continuar	Ø Ø 1 1 Ø Ø Ø Ø 3Ø ←— d —→	12 si C = Ø 7 si C = 1	
JR Z, d	Si Z = 1 PC ← PC + d Si no continuar	Ø Ø 1 Ø 1 Ø Ø Ø 28 ←— d —→	12 si Z = 1 7 si Z = Ø	
JR NZ, d	Si Z = Ø PC ← PC + d Si no continuar	Ø Ø 1 Ø Ø Ø Ø Ø 2Ø ←— d —→	12 si Z = Ø 7 si Z = 1	
JP (HL)	Bifurcación indirecta HL: PC ← HL	1 1 1 Ø 1 Ø Ø 1 E9	4	
JP (IX)	Bifurcación indirecta IX : PC ← IX	1 1 Ø 1 1 1 Ø 1 DD 1 1 1 Ø 1 Ø Ø 1 E9	8	
JP (IY)	Bifurcación indirecta IY : PC ← IY	1 1 1 1 1 1 Ø 1 FD 1 1 1 Ø 1 Ø Ø 1 E9	8	
DJNZ d	B ← B − 1 Decrementar B si B ≠ Ø PC ← PC + d Si no continuar	Ø Ø Ø 1 Ø Ø Ø Ø 1Ø ←— d —→	13 si b ≠ Ø 8 si b = Ø	

La instrucción DJNZ es una instrucción de alto nivel que permite fácilmente programar bucles de un programa y simular así la instrucción FOR... NEXT del Basic. El registro B sirve entonces de contador de bucle. Debe ser inicializado a la cantidad de reiteraciones deseadas (número comprendido entre ∅ y 255, el número ∅ representa 256 reiteraciones).

La instrucción DJNZ tiene la función de la instrucción NEXT B cuando el paso es igual a − 1. Esta decrementa el registro B y salta al principio del bucle si B es distinto de cero. Si no termina el bucle, pasando a la instrucción siguiente.

```
3∅∅       LD B,n      ;inicialización del contador de bucle.
3∅2       .
          .
          . cuerpo del bucle
          .
          .
          .
312       DJNZ $ − 1∅H; retorno al principio del bucle si B ≠ ∅
                        (dirección 3∅2)
dirección mnemotécnica
```

El símbolo cc de la tabla precedente designa una de las condiciones siguientes:

Condición		*cc*
NZ : no es cero	Z = ∅	∅ ∅ ∅
Z : cero	Z = 1	∅ ∅ 1
NC : no hay Carry	C = ∅	∅ 1 ∅
C : Carry	C = 1	∅ 1 1
PO : paridad impar	P/V = ∅	1 ∅ ∅
PE : paridad par	P/V = 1	1 ∅ 1
P : signo positivo	S = ∅	1 1 ∅
M : signo negativo	S = 1	1 1 1

1.6.11. Subprogramas

El Z 8∅ dispone de una instrucción de llamada a subprograma CALL y de una instrucción de retorno de subprograma RET que tienen respectivamente el papel de las instrucciones GOSUB y RETURN del Basic.

La instrucción CALL se resume por las operaciones siguientes:

SP ← SP − 1
(SP) ← octeto de más peso de PC
SP ← SP − 1
(SP) ← octeto de menos peso de PC

} Puesta en el stack de la dirección de retorno que está contenida en PC.

PC ← nn

} Bifurcación a la dirección del subprograma (nn).

La instrucción RET se resume por las siguientes operaciones:

Octeto de menos peso de PC ← (SP)
SP ← SP + 1
Octeto de más peso de PC ← (SP)
SP ← SP + 1

} Desapilamiento en la dirección de retorno que se coloca en el registro PC.

Mnemotécnico	*Operación*	*Código de la instrucción*	*Número de ciclos*	*Indicadores*
CALL nn	Llamada del subprograma situado en la dirección nn.	1 1 Ø Ø 1 1 Ø 1 CD ← n: menos peso → ← n: más peso →	17	
CALL cc, nn	Si cc es verdad, llamada del subprograma situado en la dirección nn. Si no continuar.	1 1 ← cc → 1 Ø Ø ← n: menos peso → ← n: más peso →	17 si cc es verdad 1Ø si cc es falso	
RET	Retorno de subprograma.	1 1 Ø Ø 1 Ø Ø 1 C9	1Ø	No se modifican
RET cc	Si cc es verdad, retorno del subprograma. Si no continuar.	1 1 ← cc → Ø Ø Ø	11 si cc es verdad 5 si cc es falso	
RST p	Llamada del subprograma situado en la dirección p.	1 1 ← t → 1 1 1	11	

El significado de cc es el mismo que en el párrafo precedente.

La instrucción RST p llama a uno de los ocho subprogramas que se dan en la tabla siguiente, que suministra igualmente la correspondencia con los tres bits del código de la instrucción.

Dirección p	*t*
ØØH	Ø Ø Ø
Ø8H	Ø Ø 1
1ØH	Ø 1 Ø
18H	Ø 1 1
2ØH	1 Ø Ø
28H	1 Ø 1
3ØH	1 1 Ø
38H	1 1 1

Estos subprogramas se sitúan en la ROM y por lo tanto no pueden ser modificados por el usuario. Son utilizados por el Basic y se procurará no emplearlos, a menos que se tomen todas las precauciones inherentes a su empleo.

Gracias a la estructura de pila utilizada podrán imbricarse varias llamadas a subprogramas; el primer RET terminará el programa más interior.

Siempre será interesante guardar los registros (PUSH) utilizados en el subprograma al principio del mismo y restaurar estos registros (POP) justo antes de la instrucción de retorno (RET). Deberá tomarse la precaución de tener tantos PUSH como POP en un subprograma con el fin de evitar que el retorno del mismo conduzca a una dirección cuyo valor sea el de un registro guardado en el stack. Este problema se debe a que la misma pila que se utiliza para los subprogramas también se utiliza para guardar los registros. Esta es la causa de numerosos errores de los principiantes que por lo general provoca un bloqueo del ordenador. En este caso se está obligado a desconectar el ordenador y volverlo a conectar. Evidentemente, el programa en memoria queda borrado. Así pues, siempre será preferible el guardar un programa ensamblado antes de comenzar su ejecución.

Ejemplo:

```
PUSH   AF
PUSH   BC guarda los registros utilizados
PUSH   DE
   .
   .
   .       cuerpo del subprograma
   .
   .
POP    DE
POP    BC restauración de los registros en orden inverso
POP    AF
RET        retorno del subprograma
```

1.6.12. Instrucciones de intercambios

Estas son instrucciones que intercambian el valor de los números de 16 bits. En la tabla siguiente la operación de intercambio se simboliza por: ↔.

Mnemotécnico	*Operación*	*Código de la instrucción*	*Número de ciclos*	*Indicadores*
EX DE, HL	DE ↔ HL	1 1 1 ∅ 1 ∅ 1 1 EB	4	No se modifican
EX AF, AF'	AF ↔ A'F'	∅ ∅ ∅ ∅ 1 ∅ ∅ ∅ ∅8	4	
EXX	BC ↔ B'C' DE ↔ D'E' HL ↔ H'L'	1 1 ∅ 1 1 ∅ ∅ 1 D9	4	
EX (SP), HL	H ↔ (SP + 1) L ↔ (SP)	1 1 1 ∅ ∅ ∅ 1 1 E3	19	
EX (SP), IX	Octeto más peso de IX ↔ (SP + 1) Octeto menos peso de IX ↔ (SP)	1 1 ∅ 1 1 1 ∅ 1 DD 1 1 1 ∅ ∅ ∅ 1 1 E3	23	
EX (SP), IY	Octeto más peso de IY ↔ (SP + 1) Octeto menos peso de IY ↔ (SP)	1 1 1 1 1 1 ∅ 1 FD 1 1 1 ∅ ∅ ∅ 1 1 E3	23	

Estas instrucciones son de uso menos importante que las precedentes, pero en ciertos casos mejoran las cualidades o posibilidades de un programa en lenguaje máquina.

1.6.13. Instrucciones de transferencia de memoria y de búsqueda

En este apartado abordaremos las instrucciones de más alto nivel del Z 8∅ que tanto han contribuido a su éxito. Las instrucciones LDI, LDIR, LDD, LDDR, permiten hacer transferencias de octetos a memoria mientras que las instrucciones CPI, CPIR, CPD y CPDR sirven para buscar un octeto en la memoria.

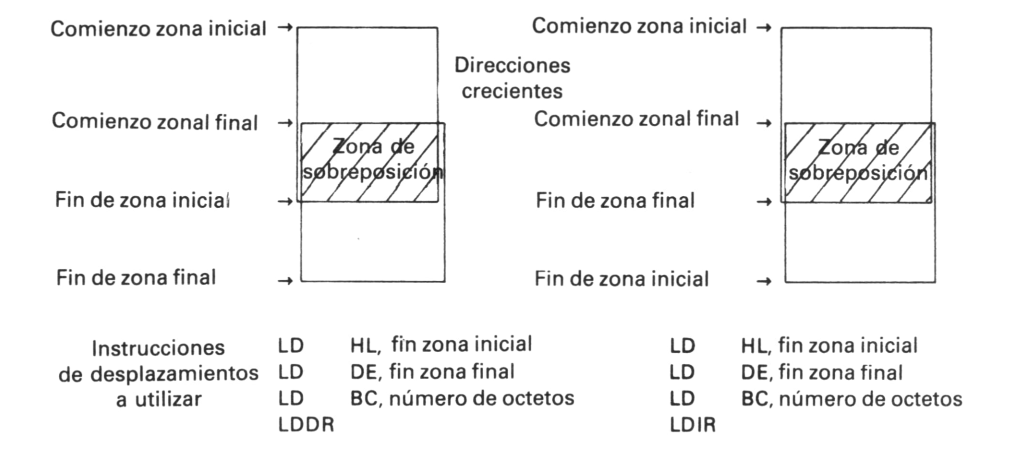
Mnemotécnico	Operación	Código de la instrucción		Número de ciclos	Indicadores
LDI	(DE) ← (HL) DE ← DE + 1 HL ← HL + 1 BC ← BC − 1	1 1 1 Ø 1 1 Ø 1 1 Ø 1 Ø Ø Ø Ø Ø	ED AØ	16	S, Z, C, no se modifican.
LDIR	(DE) ← (HL) DE ← DE + 1 HL ← HL + 1 BC ← BC − 1 Repetir mientras BC ≠ Ø	1 1 1 Ø 1 1 Ø 1 1 Ø 1 1 Ø Ø Ø Ø	ED BØ	16 + 21 * (BC − 1)	H y N puestos a cero. P/V puesto a cero para LDIR y LDDR.
LDD	(DE) ← (HL) DE ← DE − 1 HL ← HL − 1 BC ← BC − 1	1 1 1 Ø 1 1 Ø 1 1 Ø 1 Ø 1 Ø Ø Ø	ED A8	16	Si no P/V vale Ø si BC = 1. (BC = Ø al final de la instrucción) y si no 1.
LDDR	(DE) ← (HL) DE ← DE − 1 HL ← HL − 1 BC ← BC − 1 Repetir mientras BC ≠ Ø	1 1 1 Ø 1 1 Ø 1 1 Ø 1 1 1 Ø Ø Ø	ED B8	16 + 21 * (BC − 1)	

La instrucción LDI transfiere el octeto situado en la dirección HL a la posición de memoria situada en la dirección DE. DE y HL son incrementados mientras que BC es decrementado.

La instrucción LDIR ejecuta la acción de la instrucción LDI en tanto que BC sea distinto de cero. BC desarrolla así la función de contador de bucle. Inicializando BC a la cantidad de octetos que se desean desplazar, HL a la dirección inicial de la zona a desplazar, DE con la dirección inicial de la zona destino y utilizando la instrucción LDIR, el número de octetos deseado se desplaza de la zona inicial hacia la zona final. La duración de esta instrucción es proporcional al número de octetos desplazados.

Las instrucciones LDD y LDDR son análogas a las instrucciones LDI y LDIR, pero realizan la transferencia en el sentido de direcciones decrecientes.

Según que la dirección inicial sea inferior o no a la dirección final, deberá utilizarse la instrucción LDDR en lugar de la instrucción LDIR si las zonas se recubren, para evitar que las zonas de recubrimiento sean mal desplazadas.



Examinemos ahora las instrucciones de búsqueda:

Mnemotécnico	*Operación*	*Código de la instrucción*	*Número de ciclos*	*Indicadores*
CPI	Comp. si A = (HL) ? HL ← HL + 1 BC ← BC − 1	1 1 1 Ø 1 1 Ø 1 ED 1 Ø 1 Ø Ø Ø Ø 1 A1	16	S contiene el signo de A − (HL)
CPIR	Comp. si A = (HL) ? HL ← HL + 1 BC ← BC − 1 Repetir mientras A ≠ (HL) et BC ≠ Ø	1 1 1 Ø 1 1 Ø 1 ED 1 Ø 1 1 Ø Ø Ø 1 B1	16 + 21∗(BC−1)	H = 1 si hay un acarreo del bit 4 durante la comparación.
CPD	Comp. si A = (HL) ? HL ← HL − 1 BC ← BC − 1	1 1 1 Ø 1 1 Ø 1 ED 1 Ø 1 Ø 1 Ø Ø 1 A9	16	P/V vale Ø si BC = 1 si no P/V vale 1. N vale 1.
CPDR	Comp. si A = (HL) ? HL ← HL − 1 BC ← BC − 1 Repetir mientras A ≠ (HL) et BC ≠ Ø	1 1 1 Ø 1 1 Ø 1 ED 1 Ø 1 1 1 Ø Ø 1 B9	16 + 21∗(BC−1)	C no se modifica

Estas instrucciones buscan un octeto cuyo valor es idéntico al de A, a partir de la dirección HL en el sentido de las direcciones crecientes (CPI y CPIR), o en sentido inverso (CPD y CPDR) repitiendo la búsqueda, mientras que BC ≠ Ø (CPIR y CPDR) o sin repetición (CPI y CPD).

Estos dos grupos de instrucciones facilitan la programación de dos problemas muy corrientes: la transferencia a memoria y la búsqueda. Estas instrucciones son, aproximadamente, dos veces más rápidas que la serie de instrucciones elementales a las que representan. Será interesante, pues, utilizarlas lo más a menudo posible para que el código generado se más eficaz.

1.6.14. Instrucciones relacionadas con las interrupciones

Una interrupción es una señal enviada a la entrada de interrupción del microprocesador que interrumpe a éste (y por lo tanto al programa que ejecutaba) para que efectúe una serie de instrucciones llamadas subprogramas de interrupción. En el ZX SPECTRUM, un reloj interno provoca una interrupción enmascarable por el microprocesador cada 2Ø milisegundos (5Ø interrupciones por segundo). El subprograma de interrupción situado en la dirección 38H hace una lectura del teclado e incrementa el reloj de tiempo real, utilizado entre otras por la instrucción PAUSE. Guarda todos los registros para no perturbar al programa interrumpido. Este último no es sensible a las interrupciones más que en el hecho de que tarda más tiempo en ejecutarse cuando es interrumpido. No obstante, el subprograma de interrupción utiliza el registro IY que contiene el valor 5C3AH. Por lo tanto, un programa en lenguaje máquina que permite las interrupciones (instrucción EI) no deberá emplear el registro IY.

Existen dos tipos de interrupción: las interrupciones no enmascarables ligadas a la entrada NMI del Z 8Ø, y las interrupciones enmascarables ligadas a la entrada INT del Z 8Ø. Estas últimas pueden ser autorizadas o prohibidas por el programa gracias a las instrucciones EI y DI.

El Z 8Ø tiene tres modos de interrupción llamados Ø, 1 y 2.

En el modo Ø, cuando hay una interrupción enmascarable el microprocesador ejecuta una de las ocho instrucciones RST, cuya dirección se presenta en el bus de datos en ese momento.

En el modo 1, que es en el que funciona el ZX SPECTRUM, cuando hay una interrupción el Z 8Ø ejecuta la instrucción RST 38H.

En el modo 2, que es el más sofisticado, la dirección del subprograma de interrupción se coloca en la posición de memoria situada en la dirección formada por el registro I (octeto de más peso) y el octeto presente en el bus de datos (octeto de menos peso). Este último octeto ha sido suministrado por el elemento que ha provocado la interrupción.

La tabla siguiente suministra todas las instrucciones del Z 8Ø relativas a las interrupciones.

Mnemotécnico	Operación	Código de la instrucción	Número de ciclos	Indicadores
DI	Prohibición de interrupciones.	1 1 1 1 Ø Ø 1 1 F3	4	No se modifican
EI	Autorización de interrupciones.	1 1 1 1 1 Ø 1 1 FB	4	
IM Ø	Paso a modo Ø de interrupción.	1 1 1 Ø 1 1 Ø 1 ED Ø 1 Ø Ø Ø 1 1 Ø 46	8	
IM 1	Paso a modo 1 de interrupción.	1 1 1 Ø 1 1 Ø 1 ED Ø 1 Ø 1 Ø 1 1 Ø 56	8	
IM 2	Paso a modo 2 de interrupción.	1 1 1 Ø 1 1 Ø 1 ED Ø 1 Ø 1 1 1 1 Ø 5E	8	
RETI	Retorno de interrupción. Fin del subprograma de interrupción.	1 1 1 Ø 1 1 Ø 1 ED Ø 1 Ø Ø 1 1 Ø 1 4D	14	
RETN	Retorno de interrupción no enmascarable.	1 1 1 Ø 1 1 Ø 1 ED Ø 1 Ø Ø Ø 1 Ø 1 45	14	

Cuando se ejecuta un subprograma de interrupción, las interrupciones están prohibidas. El retorno al programa interrumpido mediante RETI o RSTN provoca la autorización de las interrupciones. Fuera de este detalle, el funcionamiento del subprograma de interrupción es análogo a un subprograma ordinario (apilamiento de la dirección de retorno en el momento de la llamada y desapilamiento de esta dirección cuando se hace el retorno).

Nota: Mediante la instrucción LD A,I o LD A,R es posible saber, en un momento dado, si las interrupciones están autorizadas. El indicador P/V está a 1 si las interrupciones están autorizadas y a Ø si no lo están.

1.6.15. Instrucciones de gestión de los ports de entrada/salida

El Z 8Ø posee 65536 ports de 8 bits destinados a gestionar a los periféricos. Estos ports pueden ser considerados como buses de 8 bits que realizan la conexión entre el microprocesador y los periféricos. Se numeran desde Ø a 65535.

Mnemotécnico	*Operación*	*Código de la instrucción*	*Número de ciclos*	*Indicadores*
IN A, (n)	Lectura en A del port de número An.	1 1 Ø 1 1 Ø 1 1 DB ← n →	11	No se modifican
IN r, (C)	Lectura en r del port de número (BC).	1 1 1 Ø 1 1 Ø 1 ED Ø 1 ← r → Ø Ø Ø	12	S = signo del dato. Z = 1 si el dato es nulo. P/V contiene la paridad. H y N puestos a Ø. C no se modifica.
OUT (n), A	Escritura de A en el port de número An.	1 1 Ø 1 Ø Ø 1 1 D3 ← n →	11	No se modifican
OUT (C), r	Escritura de r en el port de número (BC).	1 1 1 Ø 1 1 Ø 1 ED Ø 1 ← r → Ø Ø 1	12	No se modifican
INI	(HL) ← <port BC> B ← B − 1 HL ← HL + 1	1 1 1 Ø 1 1 Ø 1 ED 1 Ø 1 Ø Ø Ø 1 Ø A2	16	S, H, P/V desconocidos Z = 1 si B = 1 al comienzo de la
IND	(HL) ← <port BC> B ← B − 1 HL ← HL − 1	1 1 1 Ø 1 1 Ø 1 ED 1 Ø 1 Ø 1 Ø 1 Ø AA	16	instrucción. N puesto a 1. C no es modificado.
INIR	(HL) ← <port BC> B ← B − 1 HL ← HL + 1 Repetir mientras B ≠ Ø	1 1 1 Ø 1 1 Ø 1 ED 1 Ø 1 1 Ø Ø 1 Ø B2	16 + 21 * (B − 1)	S, H, P/V desconocidos Z y N son puestos a 1
INDR	(HL) ← <port BC> B ← B − 1 HL ← HL − 1 Repetir mientras B ≠ Ø	1 1 1 Ø 1 1 Ø 1 ED 1 Ø 1 1 1 Ø 1 Ø BA	16 + 21 * (B − 1)	C no es modificado
OUTI	<port BC> ← (HL) B ← B − 1 HL ← HL + 1	1 1 1 Ø 1 1 Ø 1 ED 1 Ø 1 Ø Ø Ø 1 1 A3	16	S, H, P/V desconocidos Z = 1 si B = 1 al comienzo de la
OUTD	<port BC> ← (HL) B ← B − 1 HL ← HL − 1	1 1 1 Ø 1 1 Ø 1 ED 1 Ø 1 Ø 1 Ø 1 1 AB	16	instrucción. N puesto a 1. C no es modificado.
OTIR	<port BC> ← (HL) B ← B − 1 HL ← HL + 1 Repetir mientras B ≠ Ø	1 1 1 Ø 1 1 Ø 1 ED 1 Ø 1 1 Ø Ø 1 1 B3	16 + 21 * (B − 1)	S, H, P/V desconocidos Z y N son puestos a 1
OTDR	<port BC> ← (HL) B ← B − 1 HL ← HL − 1 Repetir mientras B ≠ Ø	1 1 1 Ø 1 1 Ø 1 ED 1 Ø 1 1 1 Ø 1 1 BB	16 + 21 * (B − 1)	C no es modificado

Como veremos en el capítulo 4, el ZX SPECTRUM utiliza el port 254 para la gestión del altavoz, del lector de cassette y del borde de la pantalla; el port 251 para la gestión de la impresora, así como otros 8 ports para la gestión del teclado.

La lectura de un port se realiza mediante la instrucción IN que transfiere su contenido a un registro de 8 bits. La escritura del port se hace mediante la instrucción OUT que transfiere el contenido de un registro hacia el port.

El Z 8Ø tiene instrucciones de lectura y de escritura múltiples que efectúan una transferencia de datos entre el port y una zona de la memoria. El registro HL sirve de puntero de memoria y el registro B de contador del número de octetos transferidos. Estas instrucciones de alto nivel son de un relativo poco interés en el SPECTRUM.

En todas las instrucciones de entrada/salida del Z 8Ø el número del port está en el registro BC, o en el A (octeto de más peso) y en un entero n (octeto de menor peso).

1.7. Instrucciones Basic relativas a la utilización del lenguaje máquina

El Basic del ZX SPECTRUM tiene seis instrucciones que permiten utilizar programas en lenguaje máquina.

La instrucción *POKE dirección, octeto* sirve para modificar un octeto de la memoria. Equivale a las dos instrucciones de ensamblador siguientes:

```
LD A, octeto
LD (dirección), A
```

La función *PEEK dirección* permite leer un octeto de la memoria. Es equivalente a:

```
LD A, (dirección)
```

El valor leído podrá ser visualizado mediante: *PRINT PEEK dirección*.

La función *USR dirección* provoca la bifurcación del Z 8Ø a la dirección especificada. Es equivalente a CALL dirección. Dado que USR es una función, deberá ser utilizada con otra instrucción como PRINT. Así la orden *PRINT USR dirección* provocará la bifurcación especificada. Si el puntero del stack no es modificado por el programa en lenguaje máquina colocado en esta dirección, la instrucción RET producirá el retorno al Basic.

La instrucción *CLEAR dirección* fija la dirección más alta utilizable por un programa Basic. Los octetos situados más allá de esta direc-

ción podrán emplearse por un programa en lenguaje máquina sin riesgo de que el programa Basic modifique estos octetos. Cuando se utilice a la vez un programa Basic y un programa en lenguaje máquina será necesario efectuar esta instrucción para evitar los conflictos que seguramente se producirán entre los dos programas.

La instrucción *SAVE «nombre» CODE dirección inicial, longitud* transfiere la zona de memoria que empieza en la dirección especificada a un fichero en cassette con el nombre especificado. Esta instrucción permite guardar los programas en lenguaje máquina.

La instrucción *LOAD «nombre» CODE* permitirá realizar la transferencia inversa para cargar en memoria el fichero cuyo nombre se especifica, el cual contiene un programa en lenguaje máquina.

2. Utiles de programación en ensamblador

En este capítulo describiremos cómo utilizar el Editor/Ensamblador y el Debugger sobre un ejemplo de programa.

Existen actualmente varios Ensambladores y varios Debuggers para el ZX SPECTRUM, pero esencialmente sólo describiremos el Editor/Ensamblador y el Debugger. No obstante, las ideas dadas en este capítulo serán válidas para otros útiles de desarrollo en Ensamblador. Solamente varía la sintaxis. Así pues, insistiremos más sobre el fin de las órdenes que sobre su sintaxis que ya se explica en los manuales de usuario.

2.1. Utilización del Editor/Ensamblador

Examinemos un programa capaz de transferir un número dado de octetos de un lugar de la memoria a otro. Este programa puede ser simplemente escrito mediante la utilización de la instrucción LDIR. Supongamos que queremos transferir 1Ø octetos situados a partir de la dirección 79ØØH hacia el espacio memoria que empieza en 7AØØH. El programa se escribirá en mnemónicos:

```
LD  HL, 79ØØH
LD  DE, 7AØØH
LD  BC, 1Ø
LDIR
```

Carguemos el Editor/Ensamblador y tecleemos el programa. El texto siguiente se visualizará en la pantalla:

```
        ZX SPECTRUM EDITOR/ENSAMBLADOR
 Autor © 1983 Pascal PELLIER y Ediciones
                EYROLES
>I
0010            LD          HL,7900H
0020            LD          DE,7A00H
0030            LD          BC,10
0040            LDIR
0050
>A
0000 210079      0010             LD
     HL,7900H
0003 11007A      0020             LD
     DE,7A00H
0006 010A00      0030             LD
     BC,10
0009 EDB0        0040             LDIR
No hay END
00001 Error (s)
00000 es la dirección de lanzamiento
```

Comentarios

- Mensaje del autor visualizado en la introducción
- Orden de inserción de texto
- Escritura del programa con utilización de caracteres de tabulación <TAB>
- Pulsar <BREAK>
- Orden de ensamblaje
- Visualización del texto ensamblado
- Visualización de un error

Después del ensamblaje, el Editor/Ensamblador visualiza para cada una de las instrucciones la dirección donde está situada, el código de la instrucción con 1, 2, 3 o 4 octetos, el número de línea correspondiente al mnemónico de la instrucción y los operandos. Así, la instrucción LD DE, 7AØØH es colocada en la dirección ØØØ3 (hexadecimal) y su código es 11ØØ7A (3 octetos).

Tal como está, el programa se colocará en la dirección cero, es decir, en memoria muerta (ROM). No podría ser ejecutado, puesto que no se le puede colocar en memoria muerta. Para colocarlo en otro lugar hay que indicar al ensamblador la dirección donde empieza con una orden ORG (origen). Una orden de ensamblaje es una pseudoinstrucción que permite actuar sobre el desarrollo del ensamblaje. Se coloca en una línea del programa en lenguaje ensamblador de la misma forma que una instrucción del Z 8Ø.

Para colocar nuestro programa en la dirección 78ØØH (memoria viva), deberá añadirse en cabeza del programa la orden: ORG 78ØØH.

```
>I5                                  Inserción en línea 5
0005            ORG         7800H
No queda espacio entre líneas
```

Cuando se ejecuta la orden de inserción el Editor intenta insertar una línea después de la última línea insertada. Para esto añade el paso de inserción (1Ø por defecto) al último número de la línea insertada. Si este número es superior al número de la próxima línea del texto no podrá realizarse la inserción y visualizará: «No hay más lugar entre

las líneas». Esto es lo que se ha producido debido a la inserción de la línea 5.

En el programa que hemos tecleado, el ensamblador señala el error «Sin END». Todo programa ensamblador debe terminar normalmente con la orden END, que indica el fin del programa y que especifica su dirección de desplazamiento, es decir, la dirección donde la ejecución debe empezar. Si nuestro programa está colocado en la 78ØØH, la dirección de lanzamiento será 78ØØH. Deberemos, pues, colocar la directiva END 78ØØH.

Si deseamos cambiar la dirección donde esté situado será necesario cambiar a la vez la orden ORG y la orden END. Es posible evitar esto empleando una etiqueta simbólica. Una etiqueta es una serie de caracteres alfanuméricos que representan una dirección de memoria. La definición de una etiqueta se hace colocándola justo después del número de línea en una instrucción. Seguidamente podremos hacer referencia al valor de la etiqueta colocándola en cualquier expresión aritmética autorizada por el ensamblador.

En este ejemplo coloquemos una etiqueta INICIO en la primera instrucción. Esta etiqueta representa la dirección de lanzamiento del programa. Por lo tanto podrá ser utilizada con la directiva END.

```
>E10
0010 INICIO   LD     HL,7900H      Modificación de la línea 1Ø para la
>I$                                etiqueta INICIO
0050          END    INICIO        Inserción de la directiva END al final
0060                               del programa
>N,10                              Renumeración del texto
>P#,$
0010          ORG    7800H
0020 INICIO   LD     HL,7900H      Visualización del texto
0030          LD     DE,7A00H
0040          LD     BC,10
0050          LDIR
0060          END    INICIO
```

El texto que acabamos de grabar se llama versión *fuente* del programa. Es posible guardarlo en cassette a través de la orden SAVE del editor.

```
> S TRANSFERT           Guarda el programa con el nombre:
Preparar el cassette    «TRANSFERT».
```

La versión en lenguaje máquina del programa o versión *objeto* puede guardarse en cassette gracias a la opción WO (Write Output) del ensamblador:

```
>AOBJET-WO
7800            0010          ORG
    7800H
7800 210079     0020 INICIO   LD
    HL,7900H
7803 11007A     0030          LD
    DE,7A00H
7806 010A00     0040          LD
    BC,10
7809 EDB0       0050          LDIR
7800            0060          END
    INICIO
00000 Error (s)
30720 es la dirección de lanzamiento

Preparar el cassette
>
```

Ensamblaje del programa y creación del objeto sobre cassette con el nombre: «OBJET»

Dirección de lanzamiento 3Ø72Ø = 78ØØH

Utilicemos ahora este programa para transferir de un lugar a otro un mensaje constituido por caracteres ASCII. El código ASCII permite presentar todos los caracteres alfanuméricos en los 7 bits menos significativos de un octeto. En el SPECTRUM, el octavo bit está a cero. Existe una correspondencia biunívoca entre un carácter y su código ASCII. La orden DEFM del ensamblador asegura la transcripción en códigos ASCII de una serie de caracteres dados entre comillas y genera los caracteres correspondientes.

Ejemplo:

```
DEFM     «Bonjour»
```

Esta orden genera sucesivamente los 7 octetos que representan a los códigos ASCII de cada una de las letras del nombre «Bonjour».

Empleemos ahora la orden EQU para dar a la etiqueta LONG la longitud del mensaje asociado a la directiva DEFM.

```
FUENTE        DEFM        «Bonjour»
LONG          EQU         $—FUENTE
```

La expresión $—FUENTE es igual a la longitud del mensaje, puesto que $ designa la dirección actual; es decir, la dirección del último octeto del mensaje más 1 y FUENTE es la dirección del primer octeto del mensaje. Este valor es colocado en la etiqueta LONG por la orden EQU. El uso de esta etiqueta hará el programa independiente de la longitud del mensaje utilizado.

Para especificar el lugar donde vamos a transferir el mensaje emplearemos la orden DEFS que permite reservar cierto número de octetos para guardar variables o tablas:

```
DEST            DEFS        LONG
```

Esta orden reserva un número igual a LONG de octetos. DEST contiene la dirección de la zona de memoria reservada.

Efectuemos estas modificaciones en el programa y examinemos el código generado.

```
>E 20
0020 INICIO       LD       HL,FUENTE          Modificación de la línea 2Ø.
0030              LD       DE,7A00H           Pasar a la línea siguiente ( ↓ )
>E
0030              LD       DE,DEST
0040              LD       BC,10              Modificación de la línea en curso
>E
0040              LD       BC,LONG
>I51,1
0051 FUENTE       DEFM     'Bonjour'          Inserción de tres directivas
0052 LONG         EQU      $-FUENTE           suplementarias
0053 DEST         DEFS     LONG
0054
>N,,10
>A-WS
7800              0010              ORG       Renumeración.
     7800H                                    Ensamblaje con visualización
7800 210B78       0020 INICIO       LD        de símbolos
     HL,FUENTE
7803 111278       0030              LD
     DE,DEST
7806 010700       0040              LD
     BC,LONG
7809 EDB0         0050              LDIR
780B 42           0060 FUENTE       DFEM
     'Bonjour'
       6F  6E  6A  6F  75  72
0007              0070 LONG         EQU       Códigos ASCII de «Bonjour».
     $-FUENTE                                 Longitud de Bonjour = 7
7812              0080 DEST         DEFS
     LONG
7800              0090              END
     INICIO
00000 Error (s)
30720 es la dirección de lanzamiento

INICIO     7800     DEST      7812            Tabla de símbolos con su valor
LONG       0007     FUENTE    780B            enfrente
>
```

Para terminar, tomaremos de nuevo nuestro programa y lo modificaremos de manera que prescindamos de la instrucción LDIR mediante una serie de 8 instrucciones que formen un bucle. La última instruc-

ción es una instrucción de salto que permite el retorno al principio del bucle simbolizado por la etiqueta LOOP.

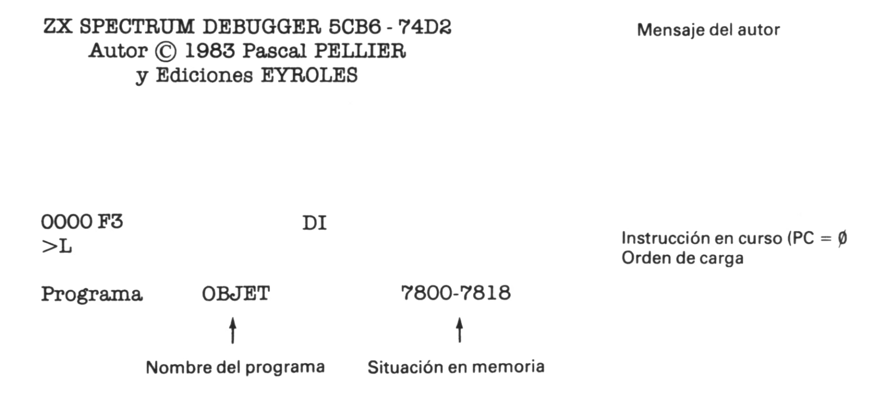
```
0010 INICIO   ORG    7800H
0020          LD     HL,FUENTE
0030          LD     DE,DEST
0040          LD     BC,LONG
0050 LOOP     LD     A,(HL)      ; lectura de un carácter
0060          LD     (DE),A      ; transferencia del carácter
0070          INC    HL          ; aumentar los punteros
0080          INC    DE
0090          DEC    BC          ; decrementar el contador de bucle
0100          LD     A,B         ; comprueba si BC es nulo
0110          OR     C
0120          JR     NZ,LOOP     ; bifurcación si la transferencia
0130 FUENTE   DEFM   'Bonjour'   ; no ha terminado
0140 LONG     EQU    $-FUENTE
0150 DEST     DEFS   LONG
0160          END    INICIO
```

En estos ejemplos nos hemos percatado de algunas de las posibilidades de edición del Editor/Ensamblador. Posee muchas más que serán de utilidad cuando se escriban programas más importantes.

Igualmente hemos visto el interés de utilizar etiquetas simbólicas y órdenes de ensamblaje. Las etiquetas permiten especificar simplemente una posición. El texto fuente podrá fácilmente ser trasladado de un lugar a otro de la memoria, mediante la orden ORG, encargándose el ensamblador de recalcular todas las direcciones de las etiquetas. Las órdenes de ensamblaje permiten modificar el desarrollo del ensamblaje (ORG, END) o solicitan la generación de constantes o espacios para las variables (DEFM, DEFB, DEFW, DEFS). La lista completa y detallada de estas órdenes se encuentra en el manual que acompaña al Editor/Ensamblador.

2.2. Utilización del Debugger

Carguemos el Debugger y la versión objeto de nuestro último programa de transferencia utilizando la orden L (LOAD) del Debugger. Esta visualiza el nombre del programa cargado y su situación en memoria (principio-fin).



El programa OBJETO está cargado en memoria; desensamblémoslo en mnemotécnicos del Z 8Ø.

```
>D7800H,7811H                              Orden de desensamblaje entre
                                           las direcciones 78ØØH y 7811H
7800 21 12 78     LD      HL,7812
7803 11 19 78     LD      DE,7819
7806 01 07 00     LD      BC,0007
7809 7E           LD      A,(HL)
780A 12           LD      (DE),A
780B 23           INC     HL
780C 13           INC     DE
780D 0B           DEC     BC
780E 78           LD      A,B
780F B1           OR      C
7810 20 F7        JR      NZ,7809
Direc. Código      Mnemotécnico
```

Nos encontramos con el texto fuente de nuestro programa, donde las etiquetas que son propias al Editor/Ensamblador han sido reemplazadas por su valor en hexadecimal.

Visualicemos esta parte de la memoria en hexadecimal y en ASCII para examinar las variables y los datos:

> M 78ØØH

```
ADRESSES      78ØØ     A 787F           ASCII
ØØ  2112      7811     1978     Ø1Ø7    !.x..x..
Ø8  ØØ7E      1223     13ØB     78B1    .".#..x.
1Ø  2ØF7      426F     6E6A     6F75    .Bonjou      Mensaje «Bonjour»
18  72ØØ      ØØØØ     ØØØØ     ØØØØ    ........
2Ø  ØØØØ      ØØØØ     ØØØØ     ØØØØ    ........
28  ØØØØ      ØØØØ     ØØØØ     ØØØØ    ........
3Ø  ØØØØ      ØØØØ     ØØØØ     ØØØØ    ........
38  ØØØØ      ØØØØ     ØØØØ     ØØØØ    ........
4Ø  ØØØØ      ØØØØ     ØØØØ     ØØØØ    ........
48  ØØØØ      ØØØØ     ØØØØ     ØØØØ    ........
5Ø  ØØØØ      ØØØØ     ØØØØ     ØØØØ    ........
58  ØØØØ      ØØØØ     ØØØØ     ØØØØ    ........
6Ø  ØØØØ      ØØØØ     ØØØØ     ØØØØ    ........
68  ØØØØ      ØØØØ     ØØØØ     ØØØØ    ........
7Ø  ØØØØ      ØØØØ     ØØØØ     ØØØØ    ........
78  ØØØØ      ØØØØ     ØØØØ     ØØØØ    ........
```

Ejecutemos ahora el programa paso a paso (instrucción tras instrucción) gracias a la orden @ del Debugger. Pero antes inicialicemos el contador ordinal (registro PC) al valor 78ØØH que es la dirección de lanzamiento de nuestro programa.

```
>RPC = 7800H

AF=0000    BC=0000    DE=0000    HL=0000      Registros primarios
'AF=0000   BC=0000    DE=0000    HL=0000      Registros secundarios
IX=0000    IY=5C3A    SP=FFFF    PC=7800      Registro de 16 bits
```

Esta orden visualiza el valor de todos los registros, la mayoría de los cuales están inicializados a cero. Los registros de 8 bits se agrupan para formar registros dobles.

Ejecutemos paso a paso nuestro programa pulsando @ para cada instrucción y examinemos el valor de los registros en el curso de la ejecución.

```
7800  21   12  78       LD    HL,7812               Primera instrucción
 AF=ØØØØ   BC=ØØØØ      DE=ØØØØ   HL=7812
'AF=ØØØØ   BC=ØØØØ      DE=ØØØØ   HL=ØØØØ
 IX=ØØØØ   IY=5C3A      SP=FFFF   PC=78Ø3
78Ø3  11   19  78       LD    DE,7819               HL es inicializado
 AF=ØØØØ   BC=ØØØØ      DE=7618   HL=7612
'AF=ØØØØ   BC=ØØØØ      DE=ØØØØ   HL=ØØØØ           DE es inicializado
 IX=ØØØØ   IY=5C3A      SP=FFFF   PC=78Ø6
78Ø6  Ø1   Ø7  ØØ       LD    BC,ØØØ7
 AF=ØØØØ   BC=ØØØ7      DE=7619   HL=7812
'AF=ØØØØ   BC=ØØØØ      DE=ØØØØ   HL=ØØØØ           BC es inicializado
 IX=ØØØØ   IY=5C3A      SP=FFFF   PC=78Ø9
78Ø9  7E                LD    A,(HL)                Inicio del bucle
 AF=42ØØ   BC=ØØØ7      DE=7819   HL=7812
'AF=ØØØØ   BC=ØØØØ      DE=ØØØØ   HL=ØØØØ
 IX=ØØØØ   IY=5C3A      SP=FFFF   PC=78ØA
78ØA  12                LD    (DE),A                Transferencia de un octeto
 AF=42ØØ   BC=ØØØ7      DE=7819   HL=7812
'AF=ØØØØ   BC=ØØØØ      DE=ØØØØ   HL=ØØØØ
 IX=ØØØØ   IY=5C3A      SP=FFFF   PC=78ØB
78ØB  23                INC   HL
 AF=42ØØ   BC=ØØØ7      DE=7819   HL=7813
'AF=ØØØØ   BC=ØØØØ      DE=ØØØØ   HL=ØØØØ
 IX=ØØØØ   IY=5C3A      SP=FFFF   PC=78ØC
78ØC  13                INC   DE
 AF=42ØØ   BC=ØØØ7      DE=781A   HL=7813
'AF=ØØØØ   BC=ØØØØ      DE=ØØØØ   HL=ØØØØ
 IX=ØØØØ   IY=5C3A      SP=FFFF   PC=78ØD
78ØD  ØB                DEC   BC
 AF=42ØØ   BC=ØØØ6      DE=781A   HL=7813
'AF=ØØØØ   BC=ØØØØ      DE=ØØØØ   HL=ØØØØ
 IX=ØØØØ   IY=5C3A      SP=FFFF   PC=78ØE
78ØE  78                LD    A,B
 AF=ØØØØ   BC=ØØØ5      DE=781A   HL=7813
'AF=ØØØØ   BC=ØØØØ      DE=ØØØØ   HL=ØØØØ
 IX=ØØØØ   IY=5C3A      SP=FFFF   PC=78ØF
78ØF  B1                DR    C
 AF=ØØØ4   BC=ØØØ6      DE=781A   HL=7813
'AF=ØØØØ   BC=ØØØØ      DE=ØØØØ   HL=ØØØØ
 IX=ØØØØ   IY=5C3A      SP=FFFF   PC=781Ø
781Ø  20   F7           JR    NZ,78Ø9               Comprueba el fin del bucle
 AF=Ø6Ø4   BC=ØØØ6      DE=781A   HL=7813
'AF=ØØØØ   BC=ØØØØ      DE=ØØØØ   HL=ØØØØ
 IX=ØØØØ   IY=5C3A      SP=FFFF   PC=78Ø9
78Ø9  7E                LD    A,(HL)                Segunda ejecución del bucle
 AF=6FØ4   BC=ØØØ6      DE=781A   HL=7813
'AF=ØØØØ   BC=ØØØØ      DE=ØØØØ   HL=ØØØØ
 IX=ØØØØ   IY=5C3A      SP=FFFF   PC=78ØA
78ØA  12                LD    (DE),A                Etc...
```

El programa se termina cuando el registro PC alcanza el valor 7812H. Entonces el valor de los registros es el siguiente:

```
 AF=0044   BC=0000   DE=7820   HL=7819
'AF=0000   BC=0000   DE=0000   HL=0000
 IX=0000   IY=5C3A   DE=FFFF   PC=7812
```

La visualización de la página de memoria que empieza en 78ØØH permite verificar que la transferencia ha sido bien hecha:

```
ADRESSES  78ØØ   A 787F          ASCII
ØØ  2112  7811   1978    Ø1Ø7    !.x..x..
Ø8  ØØ7E  1223   13ØB    78B1    .".#..x.
1Ø  2ØF7  426F   6E6A    6F75    .Bonjou
18  7242  6F6E   6A6F    7572    .Bonjour
2Ø  ØØØØ  ØØØØ   ØØØØ    ØØØØ    ........
28  ØØØØ  ØØØØ   ØØØØ    ØØØØ    ........
3Ø  ØØØØ  ØØØØ   ØØØØ    ØØØØ    ........
38  ØØØØ  ØØØØ   ØØØØ    ØØØØ    ........
4Ø  ØØØØ  ØØØØ   ØØØØ    ØØØØ    ........
48  ØØØØ  ØØØØ   ØØØØ    ØØØØ    ........
5Ø  ØØØØ  ØØØØ   ØØØØ    ØØØØ    ........
58  ØØØØ  ØØØØ   ØØØØ    ØØØØ    ........
6Ø  ØØØØ  ØØØØ   ØØØØ    ØØØØ    ........
68  ØØØØ  ØØØØ   ØØØØ    ØØØØ    ........
7Ø  ØØØØ  ØØØØ   ØØØØ    ØØØØ    ........
78  ØØØØ  ØØØØ   ØØØØ    ØØØØ    ........
```

Duplicado del mensaje «Bonjour»

Es posible ejecutar el programa normalmente utilizando la orden G (GO) que permite lanzar la ejecución a partir de una dirección dada por el usuario. Si se efectúa la orden G 78ØØH directamente, el programa se ejecutará, ya que el microprocesador ejecutará todos los octetos que se encuentran después del programa en memoria (en principio éstos son octetos nulos: instrucción NOP). Cuando el registro PC haya conseguido el valor FFFFH, volverá a la dirección Ø, lo que provocará una reinicialización del ZX SPECTRUM y un borrado de la memoria.

Para evitar este inconveniente hay que colocar un punto de paro al final del programa con la letra B (Break point). Un punto de paro, de hecho, es una instrucción de salto al Debugger, constituida por tres octetos que se colocan en la dirección especificada por el usuario. Esta instrucción se coloca en el momento de la orden G y se suprime después del retorno al Debugger, de manera que su uso sea invisible para el usuario.

En nuestro ejemplo coloquemos el punto de paro número cero (pueden utilizarse hasta 1Ø puntos de paro) en la dirección 7812H mediante la orden:

```
BØ = 7812H
↑
número de punto de paro
```

Ejecutemos de una sola vez el programa mediante la orden:

```
G78ØØH
```

Hay retorno al Debugger con la visualización del valor de los registros, que es la misma que anteriormente.

Examinemos la página de memoria situada en 78ØØH para ver si la transferencia es correcta:

```
ADRESSES    78ØØ     A 787F          ASCII
ØØ  2112    7811     1978    Ø1Ø7    !.x..x..
Ø8  ØØ7E    1223     13ØB    78B1    .".#..x.
1Ø  2ØF7    426F     6E6A    6F75    .Bonjou
18  72ØØ    Ø065     6A6F    7572    ...£jour
2Ø  ØØØØ    ØØØØ     ØØØØ    ØØØØ    ........
28  ØØØØ    ØØØØ     ØØØØ    ØØØØ    ........
3Ø  ØØØØ    ØØØØ     ØØØØ    ØØØØ    ........
38  ØØØØ    ØØØØ     ØØØØ    ØØØØ    ........
4Ø  ØØØØ    ØØØØ     ØØØØ    ØØØØ    ........
48  ØØØØ    ØØØØ     ØØØØ    ØØØØ    ........
5Ø  ØØØØ    ØØØØ     ØØØØ    ØØØØ    ........
58  ØØØØ    ØØØØ     ØØØØ    ØØØØ    ........
6Ø  ØØØØ    ØØØØ     ØØØØ    ØØØØ    ........
68  ØØØØ    ØØØØ     ØØØØ    ØØØØ    ........
7Ø  ØØØØ    ØØØØ     ØØØØ    ØØØØ    ........
78  ØØØØ    ØØØØ     ØØØØ    ØØØØ    ........
```

Los cuatro últimos octetos del nombre «Bonjour» se han transferido correctamente, mientras que los tres primeros han tomado los valores hexadecimales CD, ØC y 65 que son de hecho los tres octetos de la instrucción de retorno al Debugger. Al estar el punto de paro colocado en la misma dirección que el mensaje «Bonjour» inicial, éstos son los valores del código de la instrucción de salto que se han utilizado para la transferencia de los tres primeros octetos. Al final de la ejecución esta instrucción de salto ha sido suprimida, lo que explica que ya no sea visible en el mensaje «Bonjour» inicial.

En la práctica habrá que tener en cuenta estos tres octetos después del empleo de los puntos de paro para evitar funcionamientos incorrectos o incluso «pérdidas» del programa.

Acabamos de ver en un ejemplo la utilización de las principales órdenes del Debugger. Posee otras, pero dejamos al lector que las experimente siguiendo el manual que se adjunta con el Debugger.

2.3. Utilización del Editor/Ensamblador con el Debugger

Con un SPECTRUM 48 K es posible utilizar el Debugger conjuntamente con el Editor/Ensamblador.

Para ello cargue el Debugger después de la conexión y colóquelo en la parte más alta de la memoria por medio de la orden PE 7ØØH.

Vuelva al monitor Basic mediante G121CH.

Cargue el Editor/Ensamblador que se coloca en la parte baja de la memoria (dirección 5E53H).

Escriba su programa o modifique un programa fuente previamente guardado en cassette.

Ensamble este programa en memoria (A-IM) colocándolo en un lugar en el que no aplaste al Debugger y que se encuentre después del texto fuente de su programa (el ensamblador visualizará el mensaje «Dirección incorrecta» si no ocurre así). Utilice la orden U del ensamblador para conocer el tamaño de la zona libre. Reste a esta cantidad algunas centenas de octetos para la tabla de símbolos. Restando el resultado de FFFFH obtendrá la dirección mínima de colocación de su programa. Entonces deberá verificar, después del ensamblaje, que no haya errores de «Dirección incorrecta» y que la última dirección sea inferior a E7ØØH.

Si se han satisfecho todas estas condiciones, podrá pasar al Debugger mediante la orden BE81EH. Será prudente guardar previamente su programa fuente en cassette.

Ejecute el programa paso a paso o con la orden GO colocando puntos de paro.

Para volver al Editor/Ensamblador, sin perder el programa fuente en memoria, ejecute la orden G5E61H cuando las interrupciones estén autorizadas (orden E del Debugger).

Así se puede realizar una especie de va y viene entre el Editor/Ensamblador y el Debugger hasta que el programa esté totalmente a punto.

Evidentemente, este procedimiento sólo es válido si su programa no es demasiado grande para que quepa todo en la memoria. No obstante, permite escribir programas harto consistentes. Será particularmente útil para aquellos que empiezan en la programación en lenguaje máquina.

3. Subprogramas de interés general

En este capítulo describiremos cierta cantidad de subprogramas que serán útiles para la mayor parte de los programas en ensamblador, tanto para juegos de acción rápida como para programas más serios.

Para cada subprograma daremos una explicación de su funcionamiento, su modo de empleo, así como el listado del ensamblaje abundantemente comentado. El subprograma siempre será colocado en la dirección cero por razones de homogeneidad. Si desea probarlo sobre su microordenador modifique su situación de manera que se cargue sobre la memoria viva (RAM) y que no aplaste al Debugger que le servirá para probarlo.

3.1. Multiplicación de números enteros: MUL

Este programa emplea un procedimiento análogo al que utilizamos cuando efectuamos una multiplicación en binario a mano. Tomemos por ejemplo la multiplicación 1Ø1Ø por 11ØØ; obtenemos una operación de este tipo:

```
   1Ø1Ø        multiplicando
 × 11ØØ        multiplicador
 -------
   ØØØØ
  ØØØØ         productos parciales
 1Ø1Ø
1Ø1Ø
 -------
1111ØØØ        resultado
```

Los cuatro productos parciales, en este ejemplo valen ØØØØ, o bien 1Ø1Ø, según que el bit correspondiente del multiplicador valga Ø o 1. El resultado se obtendrá realizando la suma de los diversos productos parciales convenientemente decalados. Unicamente hemos utilizado operaciones de comprobación, de sumas y de decalaje, que el microprocesador sabe efectuar. El algoritmo utilizado en el programa se deriva de esto que acabamos de decir, procediendo siempre en

sentido inverso por razones de comodidad. Se empieza por tratar los bits de más peso del multiplicador y se decala cada vez el resultado. Si el bit del multiplicador está a 1 se le añade el multiplicando; de lo contrario no se hace nada. La operación se termina cuando el bit de menos peso ha sido tratado.

```
               00010 ;***********************************************************************
               00020 ;
               00030 ;  SUBPROGRAMA DE MULTIPLICACION ENTERA DE UN
               00040 ;  NUMERO DE 16 BITS POR UN NUMERO DE 8 BITS
               00050 ;
               00060 ;  ENTRADA: DE CONTIENE EL MULTIPLICANDO
               00070 ;           A CONTIENE EL MULTIPLICADOR
               00080 ;
               00090 ;  SALIDA:  HL CONTIENE EL RESULTADO DE LA MULTIPLICACION
               00100 ;
               00110 ;
               00120 ;***********************************************************************
0000 210000    00130        LD    HL,0    ; INICIALIZACION DEL RESULTADO
0003 0608      00140        LD    B,8     ; NUMERO DE BITS A MULTIPLICAR
0005 29        00150 H0     ADD   HL,HL   ; EL RESULTADO ES MULTIPLICADO POR 2
0006 CB27      00160        SLA   A       ; EL BIT DE MAS PESO DE A ES
               00170                      ; PUESTO EN EL CARRY
0008 3001      00180        JR    NC,H1   ; SI ESTE BIT VALE Ø
000A 19        00190        ADD   HL,DE   ; SI NO AÑADIR EL MULTIPLICANDO
000B 10F8      00200 H1     DJNZ  H0      ; TRATAR TODOS LOS BITS
000D C9        00210        RET           ; FIN DEL SUBPROGRAMA
0000           00220        END
00000 TOTAL ERRORES
```

El programa efectúa una multiplicación de un número de 16 bits por un número de 8 bits y da un resultado de 16 bits. Así, si los números son demasiado grandes, puede que el resultado no quepa en 16 bits, entonces hay un desbordamiento y el resultado es erróneo. Por consiguiente, habrá que asegurarse que no ocurra esto cuando se emplee esta rutina de multiplicación.

3.2. División de números enteros: DIV

El algoritmo utilizado en estos subprogramas proviene del procedimiento empleado cuando se efectúa una división binaria a mano. Tomemos como ejemplo la división 1Ø11Ø11 por 111Ø:

```
       dividendo    1Ø11Ø11 | 111Ø      divisor
                            |---------
restos parciales    1Ø11Ø   | Ø11Ø      cociente
                     1ØØØ1  |
           resto      ØØ111 |
```

El cociente vale 11Ø y el resto 111.

El procedimiento utilizado en el programa es el siguiente: el registro A destinado a recibir los restos parciales es inicializado a cero.

Se pone el bit de más peso del dividendo en el bit Ø de A. Se mira entonces si A es superior al divisor. Si no lo es, se coloca un bit Ø en el

cociente, si no se coloca un bit 1 y se sustrae el divisor de A. Seguidamente A es decalado hacia la izquierda y se reemprende el proceso hasta que todos los bits del dividendo sean tratados.

Ejemplo:

	Valor de A		Cociente
ciclo ∅	∅		
ciclo 1	1	∅	A es inferior al divisor
ciclo 2	1∅	∅	
ciclo 3	1∅1	∅	
ciclo 4	1∅11	∅	
ciclo 5	1∅11∅	1	A es superior al divisor
ciclo 6	1∅∅∅1	1	
ciclo 7	111	∅	A es inferior al divisor

El valor del cociente, leído de arriba abajo, es igual a ∅∅∅∅11∅ y el resto está contenido en A (111).

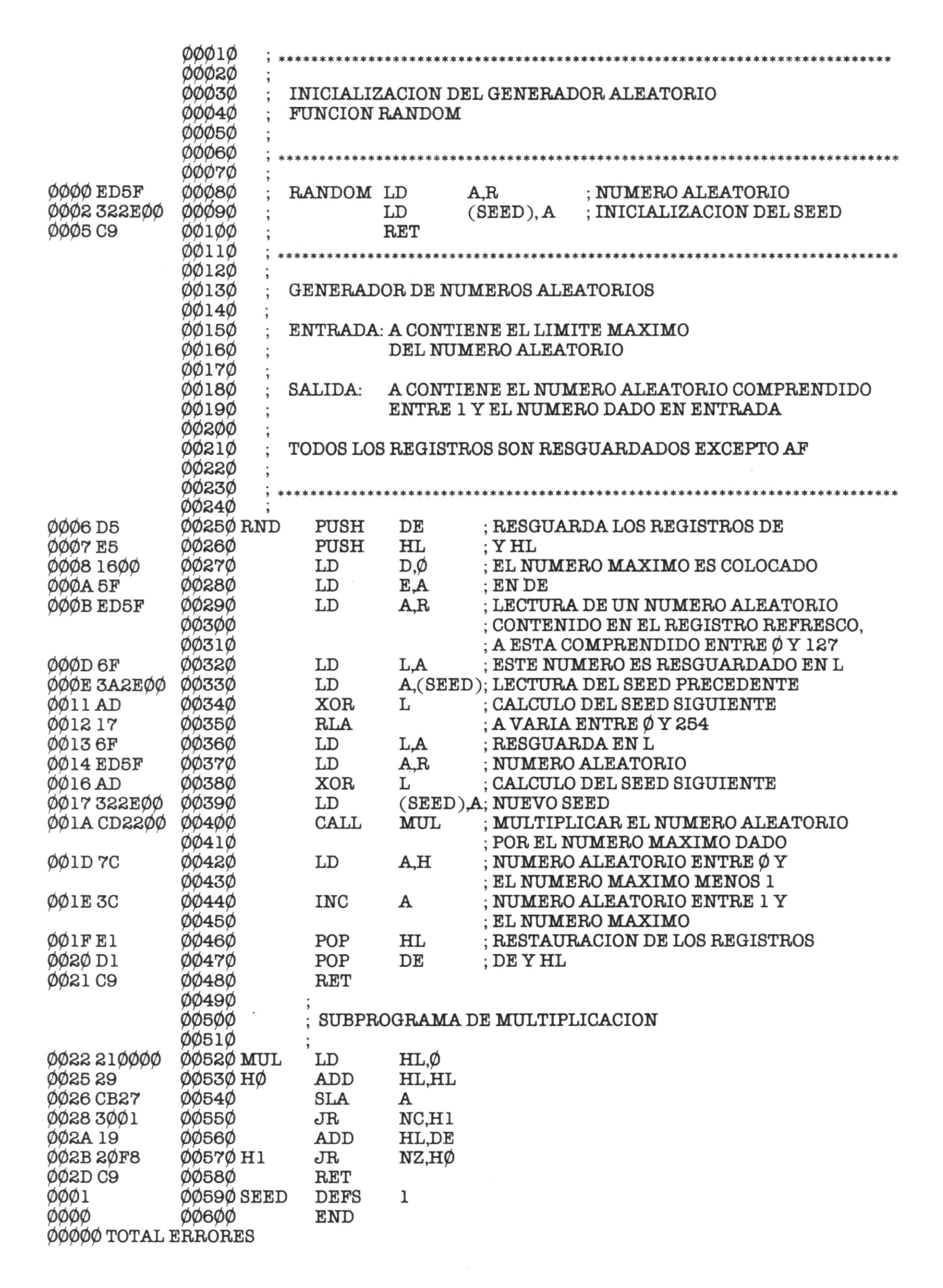
```
                    ∅∅∅1∅  ;*********************************************************************
                    ∅∅∅2∅  ;
                    ∅∅∅3∅  ;  SUBPROGRAMA DE DIVISION ENTERA DE UN
                    ∅∅∅4∅  ;  NUMERO DE 16 BITS POR UN NUMERO DE 8 BITS
                    ∅∅∅5∅  ;
                    ∅∅∅6∅  ;  ENTRADA: HL CONTIENE EL DIVIDENDO
                    ∅∅∅7∅  ;           C CONTIENE EL DIVISOR
                    ∅∅∅8∅  ;
                    ∅∅∅9∅  ;  SALIDA:  HL CONTIENE EL COCIENTE
                    ∅∅1∅∅  ;           A CONTIENE EL RESTO
                    ∅∅11∅  ;
                    ∅∅12∅  ;*********************************************************************
                    ∅∅13∅  ;
∅∅∅∅ AF             ∅∅14∅ DIV  XOR   A       ; A CONTENDRA LOS PRIMEROS BITS
                    ∅∅15∅                    ; DEL DIVIDENDO
∅∅∅1 ∅61∅           ∅∅16∅      LD    B,16    ; NUMERO DE BITS DEL DIVIDENDO
∅∅∅3 29             ∅∅17∅ H2   ADD   HL,HL   ; EL BIT 16 DEL DIVIDENDO ES
                    ∅∅18∅                    ; COLOCADO EN EL CARRY
∅∅∅4 17             ∅∅19∅      RLA           ; ESTE BIT ES COLOCADO EL DE
                    ∅∅2∅∅                    ; MENOR PESO DE A
∅∅∅5 38∅3           ∅∅21∅      JR    C,H4    ; SI HAY DESBORDAMIENTO DE A, A ES
                    ∅∅22∅                    ; SUPERIOR AL DIVISOR
∅∅∅7 B9             ∅∅23∅      CP    C       ; ES SUPERIOR A AL DIVISOR?
∅∅∅8 38∅2           ∅∅24∅      JR    C,H3    ; NO, PASAR AL BIT SIGUIENTE
∅∅∅A 91             ∅∅25∅ H4   SUB   C       ; SUSTRAER EL DIVISOR
∅∅∅B 2C             ∅∅26∅      INC   L       ; 1 EN EL COCIENTE
∅∅∅C 1∅F5           ∅∅27∅ H3   DJNZ  H2      ; BUCLE PARA TRATAR TODOS LOS BITS
∅∅∅E C9             ∅∅28∅      RET           ; FIN DEL SUBPROGRAMA
∅∅∅∅                ∅∅29∅      END
∅∅∅∅∅ TOTAL ERRORES
```

El programa efectúa una división de un número de 16 bits por un número de 8 bits, coloca el cociente en un número de 16 bits y el resto en un número de 8 bits. El único caso en el que puede producirse un desbordamiento es cuando el divisor (C) es nulo.

3.3. Generador de números aleatorios: RND

Este subprograma calcula un número aleatorio de 8 bits cuyo valor está comprendido entre 1 y un número máximo suministrado cuando se realiza la llamada al subprograma. Este empieza por determinar un número aleatorio comprendido entre ∅ y 255, utilizando el registro REFRESH del Z 8∅ y la variable SEED modificada a cada

llamada. El registro R contiene un número pseudoaleatorio comprendido entre Ø y 127; de este modo hay que efectuar un pequeño tratamiento para obtener un número que varíe entre Ø y 255. Conocido este número se le multiplica por el número máximo dado al entrar. Guardando solamente el octeto de más peso del resultado, al que se le añade 1, obtenemos el número aleatorio deseado. La función RANDOM debe ser llamada al principio del programa para inicializar la variable SEED.



```
               ØØØ1Ø  ; ************************************************************
               ØØØ2Ø  ;
               ØØØ3Ø  ;   INICIALIZACION DEL GENERADOR ALEATORIO
               ØØØ4Ø  ;   FUNCION RANDOM
               ØØØ5Ø  ;
               ØØØ6Ø  ; ************************************************************
               ØØØ7Ø  ;
ØØØØ ED5F      ØØØ8Ø  ;   RANDOM LD    A,R          ; NUMERO ALEATORIO
ØØØ2 322EØØ    ØØØ9Ø  ;          LD    (SEED), A    ; INICIALIZACION DEL SEED
ØØØ5 C9        ØØ1ØØ  ;          RET
               ØØ11Ø  ; ************************************************************
               ØØ12Ø  ;
               ØØ13Ø  ;   GENERADOR DE NUMEROS ALEATORIOS
               ØØ14Ø  ;
               ØØ15Ø  ;   ENTRADA: A CONTIENE EL LIMITE MAXIMO
               ØØ16Ø  ;            DEL NUMERO ALEATORIO
               ØØ17Ø  ;
               ØØ18Ø  ;   SALIDA:  A CONTIENE EL NUMERO ALEATORIO COMPRENDIDO
               ØØ19Ø  ;            ENTRE 1 Y EL NUMERO DADO EN ENTRADA
               ØØ2ØØ  ;
               ØØ21Ø  ;   TODOS LOS REGISTROS SON RESGUARDADOS EXCEPTO AF
               ØØ22Ø  ;
               ØØ23Ø  ; ************************************************************
               ØØ24Ø  ;
ØØØ6 D5        ØØ25Ø RND   PUSH  DE          ; RESGUARDA LOS REGISTROS DE
ØØØ7 E5        ØØ26Ø       PUSH  HL          ; Y HL
ØØØ8 16ØØ      ØØ27Ø       LD    D,Ø         ; EL NUMERO MAXIMO ES COLOCADO
ØØØA 5F        ØØ28Ø       LD    E,A         ; EN DE
ØØØB ED5F      ØØ29Ø       LD    A,R         ; LECTURA DE UN NUMERO ALEATORIO
               ØØ3ØØ                         ; CONTENIDO EN EL REGISTRO REFRESCO,
               ØØ31Ø                         ; A ESTA COMPRENDIDO ENTRE Ø Y 127
ØØØD 6F        ØØ32Ø       LD    L,A         ; ESTE NUMERO ES RESGUARDADO EN L
ØØØE 3A2EØØ    ØØ33Ø       LD    A,(SEED)    ; LECTURA DEL SEED PRECEDENTE
ØØ11 AD        ØØ34Ø       XOR   L           ; CALCULO DEL SEED SIGUIENTE
ØØ12 17        ØØ35Ø       RLA               ; A VARIA ENTRE Ø Y 254
ØØ13 6F        ØØ36Ø       LD    L,A         ; RESGUARDA EN L
ØØ14 ED5F      ØØ37Ø       LD    A,R         ; NUMERO ALEATORIO
ØØ16 AD        ØØ38Ø       XOR   L           ; CALCULO DEL SEED SIGUIENTE
ØØ17 322EØØ    ØØ39Ø       LD    (SEED),A    ; NUEVO SEED
ØØ1A CD22ØØ    ØØ4ØØ       CALL  MUL         ; MULTIPLICAR EL NUMERO ALEATORIO
               ØØ41Ø                         ; POR EL NUMERO MAXIMO DADO
ØØ1D 7C        ØØ42Ø       LD    A,H         ; NUMERO ALEATORIO ENTRE Ø Y
               ØØ43Ø                         ; EL NUMERO MAXIMO MENOS 1
ØØ1E 3C        ØØ44Ø       INC   A           ; NUMERO ALEATORIO ENTRE 1 Y
               ØØ45Ø                         ; EL NUMERO MAXIMO
ØØ1F E1        ØØ46Ø       POP   HL          ; RESTAURACION DE LOS REGISTROS
ØØ2Ø D1        ØØ47Ø       POP   DE          ; DE Y HL
ØØ21 C9        ØØ48Ø       RET
               ØØ49Ø       ;
               ØØ5ØØ       ; SUBPROGRAMA DE MULTIPLICACION
               ØØ51Ø       ;
ØØ22 21ØØØØ    ØØ52Ø MUL   LD    HL,Ø
ØØ25 29        ØØ53Ø HØ    ADD   HL,HL
ØØ26 CB27      ØØ54Ø       SLA   A
ØØ28 3ØØ1      ØØ55Ø       JR    NC,H1
ØØ2A 19        ØØ56Ø       ADD   HL,DE
ØØ2B 2ØF8      ØØ57Ø H1    JR    NZ,HØ
ØØ2D C9        ØØ58Ø       RET
ØØØ1           ØØ59Ø SEED  DEFS  1
ØØØØ           ØØ6ØØ       END
ØØØØØ TOTAL ERRORES
```

3.4. Conversión de un número binario entero en una serie de caracteres ASCII: TRF

Este subprograma será muy útil para visualizar las puntuaciones en la pantalla de un programa de juego. Cada vez que la puntuación sea modificada se le llamará para reescribir la nueva puntuación, esta rutina emplea un procedimiento simple, pero que tiene la ventaja de ser muy rápido. Determina sucesivamente todas las cifras del número a traducir, comenzando por la cifra más significativa. Esto se obtiene dividiendo sucesivamente el número por todas las potencias de 1Ø, empezando por 1ØØØØ, llegando hasta 1Ø. El cociente transformado en ASCII da una de las cifras deseadas, el resto se utiliza como nuevo argumento para efectuar las otras divisiones.

```
Número = cifra 1 * 1ØØØØ + resto 1
resto 1  = cifra 2 * 1ØØØ  + resto 2
resto 2  = cifra 3 * 1ØØ   + resto 3
resto 3  = cifra 4 * 1Ø    + resto 4
resto 4  = cifra 5
```

Las cifras así calculadas se colocan las unas detrás de las otras en una zona reservada llamada BUF. Después de la última división se ordena el número de unidades, se coloca un cero al final de la cadena y se posiciona HL al principio. Entonces se puede utilizar el subprograma MES para la visualización en la pantalla (véase cap. 4).

```
ØØØ1Ø  ;**********************************************************************************
ØØØ2Ø  ;
ØØØ3Ø  ;  TRANSFORMACION DE UN NUMERO ENTERO CON SIGNO SOBRE 16
ØØØ4Ø  ;  BITS. EN UNA SERIE DE CARACTERES ASCII QUE LO REPRESENTAN.
ØØØ5Ø  ;  EL NUMERO PUEDE VARIAR ENTRE -32768 y 32767
ØØØ6Ø  ;
ØØØ7Ø  ;  PARA OBTENER UNA CONVERSION SOBRE UN ENTERO SIN SIGNO
ØØØ8Ø  ;  CUYO VALOR ESTA COMPRENDIDO ENTRE Ø y 65535 ES
ØØØ9Ø  ;  SUFICIENTE DESTRUIR LAS LINEAS 23Ø A 3ØØ
ØØ1ØØ  ;
ØØ11Ø  ;  ENTRADA: HL CONTIENE EL NUMERO A CONVERTIR
ØØ12Ø  ;
ØØ13Ø  ;  SALIDA:    HL CONTIENE LA DIRECCION DEL PRIMER
ØØ14Ø  ;             ELEMENTO DE LA CADENA QUE REPRESENTA
ØØ15Ø  ;             AL NUMERO. ASI ES POSIBLE UTILIZAR
ØØ16Ø  ;             EL SUBPROGRAMA MES PARA LA VISUALIZACION
ØØ17Ø  ;
ØØ18Ø  ;**********************************************************************************
```

```
                         ØØ19Ø ;
ØØØØ DD215ØØØ            ØØ2ØØ TRF  LD    IX,BUF+1  ; IX SIRVE PARA LLENAR EL BUFFER
                         ØØ21Ø                      ; QUE CONTENDRA LA CADENA
ØØØ4 DD36FF20            ØØ22Ø      LD    (IX-1),20H; NO HAY SIGNO POR DEFECTO
ØØØ8 CB7C                ØØ23Ø      BIT   7,H       ; ¿EL NUMERO ES NEGATIVO?
ØØØA 28ØB                ØØ24Ø      JR    Z,TRD     ; SI NO
ØØØC 11ØØØØ              ØØ25Ø      LD    DE,Ø      ; PARA SUSTRAER EL HL DE Ø
ØØØF EB                  ØØ26Ø      EX    DE,HL     ;
ØØ1Ø B7                  ØØ27Ø      OR    A         ; PUESTA A Ø DEL CARRY
ØØ11 ED52                ØØ28Ø      SBC   HL,DE     ; HACER Ø-HL PARA OBTENER EL
                         ØØ29Ø                      ; VALOR ABSOLUTO DE HL
ØØ13 DD36FF2D            ØØ3ØØ      LD    (IX-1),'-' ; PONER UN SIGNO MENOS EN CABEZA
ØØ17 111Ø27              ØØ31Ø TRD  LD    DE,1ØØØØ  ;
ØØ1A CD3DØØ              ØØ32Ø      CALL  CAR       ; GUARDAR EL NUMERO DE DECENAS
                         ØØ33Ø                      ; DE MIL
ØØ1D 11E8Ø3              ØØ34Ø      LD    DE,1ØØØ   ;
ØØ2Ø CD3DØØ              ØØ35Ø      CALL  CAR       ; GUARDAR EL NUMERO DE LOS MILLARES
ØØ23 1164ØØ              ØØ36Ø      LD    DE,1ØØ    ;
ØØ26 CD3DØØ              ØØ37Ø      CALL  CAR       ; GUARDAR EL NUMERO DE LAS CENTENAS
ØØ29 11ØAØØ              ØØ38Ø      LD    DE,1Ø     ;
ØØ2C CD3DØØ              ØØ39Ø      CALL  CAR       ; GUARDAR EL NUMERO DE LAS UNIDADES
ØØ2F 7D                  ØØ4ØØ      LD    A,L       ; L CONTIENE EL NUMERO DE LAS UNIDADES
ØØ3Ø C63Ø                ØØ41Ø      ADD   A,3ØH     ; TRANSFORMAR EN ASCII
ØØ32 DD77ØØ              ØØ42Ø      LD    (IX+Ø),A  ; PUESTA EN EL BUFFER
ØØ35 DD36Ø1ØØ            ØØ43Ø      LD    (IX+1),Ø  ; PONER UN Ø AL FINAL
ØØ39 214FØØ              ØØ44Ø      LD    HL,BUF    ; HL APUNTA HACIA EL PRIMERO
                         ØØ45Ø                      ; ELEMENTO
ØØ3C C9                  ØØ46Ø      RET
                ØØ47Ø        ; *************************************************************************
                ØØ48Ø        ;
                ØØ49Ø        ;  SUBPROGRAMA QUE DIVIDE HL POR DE COLOCANDO
                ØØ5ØØ        ;  EL COCIENTE +3ØH EN EL BUFFER
                ØØ51Ø        ;
                ØØ52Ø        ; *************************************************************************
                ØØ53Ø        ;
ØØ3D B7         ØØ54Ø CAR  OR    A          ; PUESTA A Ø DEL CARRY
ØØ3E Ø6FF       ØØ55Ø      LD    B,255      ; B=-1
ØØ4Ø ED52       ØØ56Ø G9   SBC   HL,DE      ; SE SUSTRAE (DE) DE (HL) HASTA QUE
                ØØ57Ø                       ; HAYA DESBORDAMIENTO
ØØ42 Ø4         ØØ58Ø      INC   B          ; INCREMENTAR EL COCIENTE
ØØ43 3ØFB       ØØ59Ø      JR    NC,G9      ; SI NO HAY DESBORDAMIENTO
ØØ45 19         ØØ6ØØ      ADD   HL,DE      ; HL CONTIENE EL RESTO
ØØ46 78         ØØ61Ø      LD    A,B        ; A CONTIENE EL COCIENTE
ØØ47 C63Ø       ØØ62Ø      ADD   A,3ØH      ; TRANSFORMACION EN ASCII
ØØ49 DD77ØØ     ØØ63Ø      LD    (IX+Ø),A   ; PUESTA EN EL BUFFER
ØØ4C DD23       ØØ64Ø      INC   IX         ; INCREMENTAR EL PUNTERO DEL BUFFER
ØØ4E C9         ØØ65Ø      RET
ØØØ7            ØØ66Ø BUF  DEFS  7          ; BUFFER CONTENIENDO LA CADENA
ØØØØ            ØØ67Ø      END
ØØØØØ TOTAL DE ERRORES
```

3.5. Conversión de un número dado bajo la forma de una serie de caracteres ASCII en un número binario: ASCBIN

Este programa efectúa la transformación inversa de la precedente. Convierte un número dado bajo forma de una serie de caracteres ASCII en su representación binaria de 16 bits.

Ejemplo:

Serie ASCII	«4 1 8»	valor binario: ØØØØØØ11Ø1ØØØ1Ø
códigos correspondientes en hexadecimal	«34 31 38»	valor hexadecimal: Ø 1 A 2

El programa siguiente utiliza el hecho de que un número decimal que se escribe xyzt en ASCII vale: (((x * 1Ø) + y) * 1Ø + z) * 1Ø + t. El algoritmo informático utilizado sale directamente de esta fórmula.

Hay que hacer notar que la multiplicación por 1Ø se realiza mediante cuatro sumas.

```
               00010    ; ************************************************************************
               00020    ;
               00030    ; Transformación de un número positivo entero
               00040    ; dado bajo forma de una serie de caracteres
               00050    ; ASCII en su valor binario sobre 16 bits
               00060    ;
               00070    ; Entrada: DE contiene la dirección de la serie
               00080    ;             de caracteres ASCII constituida por
               00090    ;             cifras y terminada por un símbolo
               00100    ;             ASCII distinto de un número
               00110    ;
               00120    ; Salida:   HL contiene el número de 16 bits resultante
               00130    ;
               00140    ;           AF, BC, DE, HL son modificados
               00150    ;
               00160    ; ************************************************************************
               00170    ;
0000 210000    00180 ASCBIN   LD    HL,0    ; Resultado nulo a priori
0003 1A        00190 H5       LD    A,(DE)  ; Lectura carácter ASCII
0004 D630      00200          SUB   '0'     ; Conversión para obtener su valor
0006 D8        00210          RET   C       ; Retorno si no es una cifra
0007 FE0A      00220          CP    10      ;
0009 D0        00230          RET   NC      ; Retorno si no es una cifra
000A 13        00240          INC   DE      ; Incrementar puntero
000B 44        00250          LD    B,H     ; Copiar HL en BC
000C 4D        00260          LD    C,L     ;
000D 29        00270          ADD   HL,HL   ; Resultado = resultado * 2
000E 29        00280          ADD   HL,HL   ; Resultado = resultado * 4
000F 09        00290          ADD   HL,BC   ; Resultado = resultado * 5
0010 29        00300          ADD   HL,HL   ; Resultado = resultado * 10
0011 0600      00310          LD    B,0     ;
0013 4F        00320          LD    C,A     ; Cifra en BC
0014 09        00330          ADD   HL,BC   ; Añadir valor de la cifra
0015 18EC      00340          JR    H5      ; Pasar a la cifra siguiente
0000           00350          END
```

4. Las entradas/salidas

En este capítulo estudiaremos la programación de las entradas/salidas, es decir, la gestión software de los diversos periféricos de que dispone el ZX SPECTRUM. Este es un punto fundamental de la programación en ensamblador ya que es el medio utilizado para visualizar los resultados y conseguir informaciones.

Examinaremos sucesivamente la interfase entre los periféricos más corrientes y la unidad central; también daremos ejemplos de los subprogramas que permiten controlar a estos periféricos. Al igual que en el capítulo anterior, estos subprogramas se situarán en la dirección cero.

4.1. La pantalla de visualización

4.1.1. Distribución de la pantalla

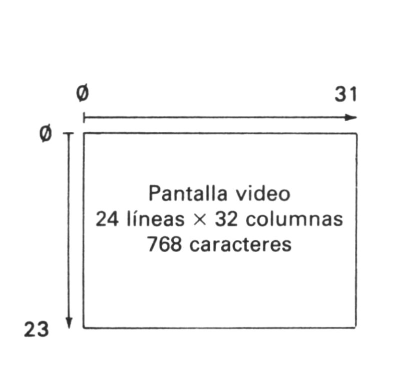
Para que puedan realizarse las operaciones de visualización, el ZX SPECTRUM posee una memoria viva de 6,75 K que sirve de interfase entre la pantalla y la unidad central. Esta memoria es leída de forma cíclica por un dispositivo hardware que genera las señales eléctricas necesarias para la visualización en pantalla de las informaciones contenidas en la memoria. Además, la escritura en esta memoria produce como consecuencia inmediata la visualización de puntos sobre la pantalla o la selección de un color.

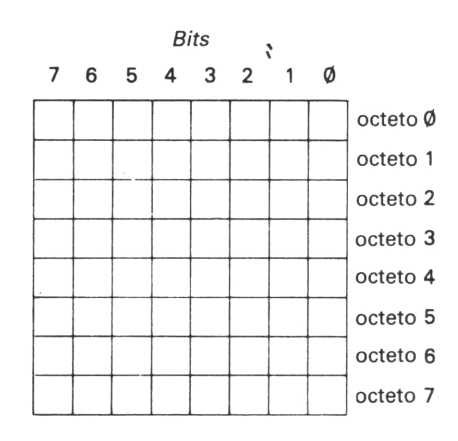
Esta memoria, a la que llamaremos *memoria video*, está constituida por dos partes:

La *primera parte* contiene la lista de los puntos de la pantalla. Ocupa 6 K y se sitúa entre las direcciones 4ØØØH y 57FFH, ambas inclusive. A cada punto de la pantalla, o *pixel*, se le asocia un bit de esta zona de memoria. Si el bit vale 1, el punto se visualiza con el color de la tinta (orden INK del Basic). Si el bit vale Ø, el punto se visualiza con el color del fondo (orden PAPER del Basic). Supongamos que el color del fondo es blanco y el color de la tinta negro. El pixel aparece

en negro sobre fondo blanco. Así, colocando algunos bits de esta zona de memoria al valor 1, provocamos la visualización de pequeños puntos negros en la pantalla.

La pantalla se divide en 24 líneas de 32 caracteres. Convencionalmente las líneas son numeradas de Ø a 23. La línea Ø está arriba de la pantalla y la línea 23 abajo. Las columnas se numeran de izquierda a derecha con los números enteros comprendidos entre Ø y 31. Cada carácter está constituido por una matriz de 8 posiciones por 8, o sea 64 posiciones. Esta matriz se almacena en memoria mediante 8 octetos. Cada octeto contiene una línea de esta matriz. Cada uno de los bits de este octeto representa una posición de la matriz. El bit más elevado contiene el punto que está más a la izquierda de la línea, el bit más bajo contiene el punto más a la derecha.





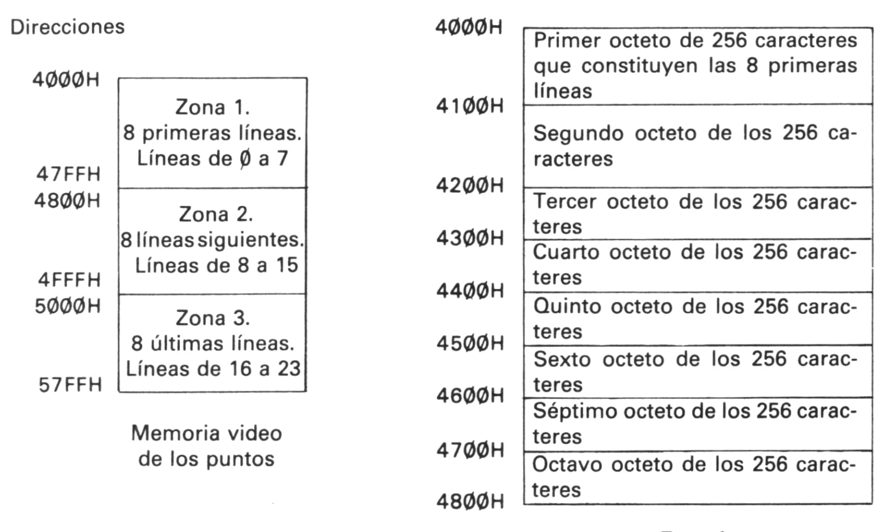
Carácter 8 × 8

La resolución del SPECTRUM es por lo tanto de 24 * 8 = 192 de 32 × 8 = 256 puntos, lo que es muy aceptable para un microordenador de esta talla.

La posición en la memoria video de los 8 octetos que contienen el carácter es menos fácil de comprender.

La memoria video que contiene los puntos se divide en tres partes de 2 K octetos, situados respectivamente en las direcciones 4ØØØH, 48ØØH y 5ØØØH. La primera parte contiene las ocho primeras líneas de la pantalla, la segunda parte las ocho siguientes y la tercera parte las ocho últimas líneas de la pantalla. En cada parte, los 256 primeros octetos contienen los primeros octetos de cada uno de los 256 caracteres (8 × 32), constituyendo las ocho líneas en el orden en el que estos caracteres son visualizados en la pantalla (los 32 primeros octetos contienen los primeros octetos de la primera línea, el primer octeto

corresponde al carácter de la izquierda de la línea). Los 256 octetos siguientes contienen los segundos octetos de los 256 caracteres que constituyen las ocho líneas y así sucesivamente hasta los 256 últimos octetos que contienen los octavos octetos de los 256 caracteres.



Zona 1

La *segunda parte* de la memoria video contiene los atributos gráficos asociados a cada carácter. Esto ocupa 768 octetos y se sitúa en las direcciones 58ØØH y 5AFFH. A cada carácter le corresponde un octeto de atributos, colocado en esta memoria en el orden en el que estos caracteres aparecen en la pantalla (los 32 primeros octetos contienen la primera línea, el primer octeto corresponde al carácter de la izquierda de la línea).

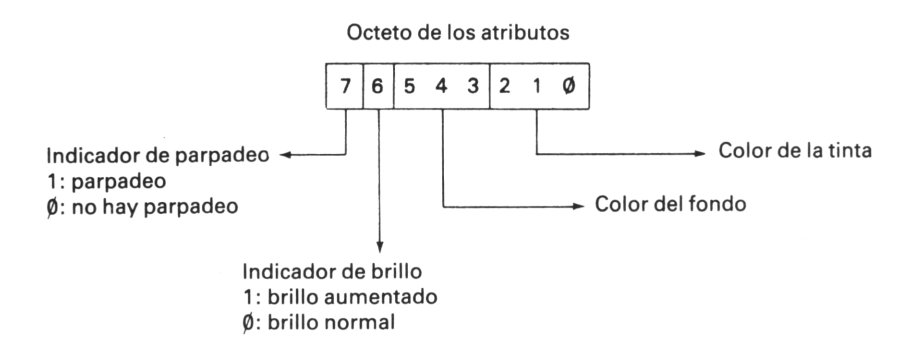
58ØØH	Atributo del carácter Ø de la línea Ø
58Ø1H	Atributo del carácter 1 de la línea Ø
.	. . .
.	. . .
.	. . .
.	. . .
5AFFH	Atributo del carácter 31 de la línea 23

Memoria video de los atributos

El ZX SPECTRUM puede visualizar ocho colores indicados con los números de Ø a 7. En consecuencia, un color puede ser memorizado en tres bits según la tabla siguiente:

Código del color	*Colores*
Ø Ø Ø	Negro
Ø Ø 1	Azul
Ø 1 Ø	Rojo
Ø 1 1	Magenta
1 Ø Ø	Verde
1 Ø 1	Cyan
1 1 Ø	Amarillo
1 1 1	Blanco

El octeto de atributos asociado a un carácter memoriza el color del fondo (color de los puntos, donde el bit asociado en memoria video es nulo) en tres bits, el color de la tinta (color de los puntos cuyo bit asociado vale 1) en tres bits, un indicador de brillo mediante 1 bit (vale 1 si el carácter tiene más brillo) y un indicador de parpadeo en 1 bit (vale 1 si el carácter parpadea).



Este procedimiento de codificación de colores relativos a un carácter limita las posibilidades gráficas del ZX SPECTRUM, puesto que no se puede fijar individualmente el color de cada uno de los puntos. No obstante, un carácter podrá colorearse rápidamente poniendo el color de fondo y el color de la tinta del color deseado, y esto sin tocar los puntos contenidos en el carácter. Este método de coloreado se utilizará en los juegos de acción rápida para el dibujo del decorado.

El último elemento programable en pantalla es el color del borde de la misma (orden BORDER del Basic). Este color, que puede ser escogido entre los ocho colores precedentes, se fija enviando el códi-

go del color al port 254. Para ello podrá utilizarse la línea de instrucciones siguientes:

```
LD      A, COLOR            ; Ø ≤ COLOR ≤ 7
OUT (254), A                ; fija el color del borde
```

El port 254 es un port cuyo número está sobre 8 bits. No hay pues que colocar el octeto de más peso de este número en el registro A.

La memoria video, que ocupa 6,75 K en memoria viva, limita la memoria viva utilizable por los programas Basic y también los programas en lenguaje máquina. Este último contiene un poco más de 9 K octetos en un SPECTRUM 16 K, lo cual no permite el funcionamiento de programas grandes. En ensamblador habrá que vigilar esta memoria video situando los programas más allá de la dirección 5BØØH. Si desea utilizar los subprogramas del interpretador Basic sitúe su programa más allá de la dirección 5CB6H.

Examinemos ahora diferentes subprogramas de visualización utilizando la organización de la pantalla tal como acabamos de describir.

4.1.2. Borrado de la pantalla: CLS

Este subprograma borra la pantalla colocando a cero todos los puntos de la memoria video y colocando a 38H (tinta negra sobre fondo blanco en brillo normal y sin destello) todos los atributos de los caracteres de la pantalla.

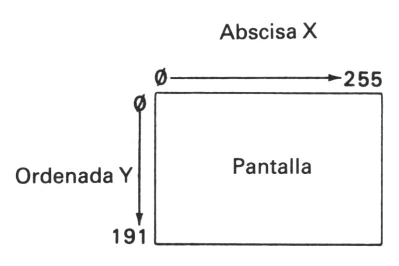
```
               00010  ;*****************************************************************************
               00020  ;
               00030  ;  BORRADO DE LA PANTALLA
               00040  ;
               00050  ;*****************************************************************************
               00060  ;
0000 210040    00070 CLS   LD    HL,4000H     ; INICIO DE LA MEMORIA DE LA PANTALLA
0003 110140    00080       LD    DE,4001H     ; INICIO DE LA MEMORIA DE LA PANTALLA +1
0006 01FF17    00090       LD    BC,17FFH     ; TAMAÑO DE LA MEMORIA DE PANTALLA -1
0009 3600      00100       LD    (HL),0       ; BORRADO DEL PRIMER CARACTER
000B EDB0      00110       LDIR               ; BORRADO DE TODA LA PANTALLA
000D 23        00120       INC   HL           ; INICIO ZONA DE ATRIBUTOS
000E 13        00130       INC   DE           ; INICIO ZONA DE ATRIBUTOS +1
000F 01FF02    00140       LD    BC,2FFH      ; LONGITUD ZONA DE ATRIBUTOS -1
0012 3638      00150       LD    (HL),38H     ; NEGRO SOBRE FONDO BLANCO
0014 EDB0      00160       LDIR               ; ESCRITURA DE TODOS LOS ATRIBUTOS
0016 C9        00170       RET
0000           00180       END
```

4.1.3. Subprogramas gráficos: SET, RESET y POINT

El subprograma SET visualiza un punto en la pantalla, RESET apaga un punto en la pantalla y POINT comprueba si un punto está presente en la misma. El punto es descrito por su abscisa, comprendi-

da entre Ø y 255 y su ordenada, comprendida entre Ø y 191. Estos tres subprogramas llaman a un mismo subprograma llamado CONV que calcula la dirección de la memoria video a la que debe accederse para realizar la función deseada. También suministra un octeto cuyo único bit a 1 corresponde al bit al que hay que acceder en la posición memoria situada en la dirección calculada.

Estos subprogramas no actúan sobre los atributos que deberán ser convenientemente posicionados para que la visualización corresponda a lo que se desea.




```
                00010  ;*************************************************************************
                00020  ;
                00030  ;  FUNCIONES GRAFICAS: SET, RESET Y PUNTO
                00040  ;
                00050  ;  ENTRADA: H CONTIENE LA ORDENADA DEL PUNTO
                00060  ;              GRAFICO 0 < =H < = 191
                00070  ;            L CONTIENE LA ABSCISA DEL PUNTO GRAFICO
                00080  ;              0 < = L < = 255
                00090  ;
                00100  ;  TODOS LOS REGISTROS SON RESGUARDADOS SALVO AF
                00110  ;
                00120  ;*************************************************************************
                00130  ;
                00140  ;  ----------------------------------------------------------------
                00150  ;
                00160  ;     ENCENDIDO DE UN PUNTO
                00170  ;
                00180  ;  ----------------------------------------------------------------
                00190  ;
0000 E5         00200 SET     PUSH   HL       ; GUARDAR POSICION
0001 CD1800     00210         CALL   CONV     ; BUSQUEDA DE LA DIRECCION DE PANTALLA
0004 B6         00220         OR     (HL)     ; PUESTA A 1 DE UN BIT
0005 77         00230         LD     (HL),A   ; VISUALIZACION
0006 E1         00240         POP    HL       ; RESTAURAR POSICION
0007 C9         00250         RET
                00260  ;  ----------------------------------------------------------------
                00270  ;
                00280  ;     DESAPARICION DE UN PUNTO
                00290  ;
                00300  ;  ----------------------------------------------------------------
                00310  ;
0008 E5         00320 RESET   PUSH   HL       ; GUARDAR POSICION
0009 CD1800     00330         CALL   CONV     ; BUSQUEDA DE LA DIRECCION DE PANTALLA
000C 2F         00340         CPL             ;
000D A6         00350         AND    (HL)     ; PUESTA A CERO DE UN BIT
000E 77         00360         LD     (HL),A   ; VISUALIZACION
000F E1         00370         POP    HL       ; RESTAURAR POSICION
0010 C9         00380         RET
```

```
                00390  ;  ------------------------------------------------------------------
                00400  ;
                00410  ;    COMPROBACION DE UN PUNTO
                00420  ;
                00430  ;  EL INDICADOR Z ESTA A 1 SI EL PUNTO ESTA APAGADO
                00440  ;  EL INDICADOR Z ESTA A Ø SI EL PUNTO ESTA ENCENDIDO
                00450  ;
                00460  ;  ------------------------------------------------------------------
                00470  ;
0011 E5         00480 PUNTO  PUSH   HL      ; GUARDAR POSICION
0012 CD1800     00490        CALL   CONV    ; BUSQUEDA DIRECCION DE PANTALLA
0015 A6         00500        AND    (HL)    ; COMPROBAR SI EL BIT ESTA A UNO
0016 E1         00510        POP    HL      ; RESTAURAR LA POSICION
0017 C9         00520        RET
                00530  ;  ------------------------------------------------------------------
                00540  ;
                00550  ;    SUBPROGRAMA DE CONVERSION
                00560  ;
                00570  ; ENTRADA: HL = ORDENADA DEL PUNTO
                00580  ;                L = ABSCISA DEL PUNTO
                00590  ;
                00600  ; SALIDA: HL = DIRECCION DEL OCTETO CORRESPONDIENTE AL PUNTO
                00610  ;              A = VALOR DEL OCTETO
                00620  ;
                00630  ;  ------------------------------------------------------------------
                00640  ;
0018 C5         00650 CONV   PUSH   BC      ; RESGUARDAR BC
0019 7C         00660        LD     A,H     ; ORDENADA DEL PUNTO
001A 2640       00670        LD     H,40H   ; INICIO PRIMERA ZONA DE LA PANTALLA
001C D640       00680        SUB    64      ; QUITAR LONGITUD DE UNA ZONA
001E 380A       00690        JR     C,ZA    ; ¿PRIMER ZONA?
0020 2648       00700        LD     H,48H   ; PRINCIPIO SEGUNDA ZONA DE LA PANTALLA
0022 D640       00710        SUB    64      ; QUITAR LONGITUD DE UNA ZONA
0024 3804       00720        JR     C,ZA    ; ¿SEGUNDA ZONA?
0026 2650       00730        LD     H,50H   ; INICIO ULTIMA ZONA DE LA PANTALLA
0028 D640       00740        SUB    64      ; QUITAR LONGITUD DE UNA ZONA
002A C640       00750 ZA     ADD    A,64    ; POSICION EN LA ZONA
002C 4F         00760        LD     C,A     ; GUARDAR ESTE VALOR EN C
002D E607       00770        AND    7       ; DECALADO VERTICAL EN EL CARACTER
002F 84         00780        ADD    A,H     ; OCTETO DE MAS PESO DE
0030 67         00790        LD     H,A     ; LA DIRECCION VIDEO
0031 79         00800        LD     A,C     ; RECUPERAR POSICION EN LA ZONA
0032 E638       00810        AND    38H     ; NUMERO DE LINEA * 8
0034 07         00820        RLCA           ; NUMERO DE LINEA * 16
0035 07         00830        RLCA           ; NUMERO DE LINEA * 32
0036 4F         00840        LD     C,A     ; GUARDAR EN C
0037 7D         00850        LD     A,L     ; ABSCISA DEL PUNTO
0038 0F         00860        RRCA           ; DIVISION POR TRES PARA OBTENER
0039 0F         00870        RRCA           ; EL NUMERO DEL CARACTER EN
003A 0F         00880        RRCA           ; LA LINEA
003B E61F       00890        AND    1FH     ; POSICION DEL CARACTER EN LA LINEA
003D 81         00900        ADD    A,C     ; AÑADIR LAS LINEAS PRECEDENTES
003E 4D         00910        LD     C,L     ; GUARDAR ABSCISA EN C
003F 6F         00920        LD     L,A     ; OCTETO DE MENOS PESO DE LA DIRECCION
0040 79         00930        LD     A,C     ; RESTURAR ABSCISA
0041 E607       00940        AND    7       ; DECALADO HORIZONTAL EN EL CARACTER
0043 3C         00950        INC    A       ; MAS UNO
0044 4F         00960        LD     C,A     ; CONTADOR DE BUCLE
0045 3E01       00970        LD     A,1     ;
0047 0F         00980 ZB     RRCA           ; DECALAR A HASTA OBTENER EL OCTETO
0048 0D         00990        DEC    C       ; QUE POSEE EL BIT A 1
0049 20FC       01000        JR     NZ,ZB   ; LA POSICION DESEADA
004B C1         01010        POP    BC      ; RESTAURAR BC
004C C9         01020        RET
0000            01030        END
```

4.1.4. Trazado de una recta: RECTA

Este programa permite que, sobre la pantalla, pueda trazarse cualquier recta definida por sus coordenadas iniciales (X_1 e Y_1) y sus coordenadas finales (X_2 e Y_2). Para explicar el algoritmo empleado supondremos que la variación de Y es inferior a la variación de X, o sea:

$$|Y_2 - Y_1| < |X_2 - X_1| \text{ y que } X_1 < X_2;\ Y_1 < Y_2$$

Se utiliza un bucle para poder dibujar sucesivamente todos los puntos de la recta. La abscisa del punto actual se incrementa a cada paso. Contrariamente, la ordenada se incrementa cada vez que es necesario para conservar la dirección de la recta. Todo el problema consiste en saber cuándo hay que incrementarla. Para ello examinaremos la ecuación de la recta:

$$Y = Y_1 + (X - X_1)\,\frac{Y_2 - Y_1}{X_2 - X_1}$$

Sea X_i la abscisa del punto actual de la recta y X_{i+1} la abscisa siguiente:

Tenemos: $X_{i+1} = X_i + 1$

y $$Y_{i+1} = Y_1 + (X_i + 1 - X_1)\,\frac{Y_2 - Y_1}{X_2 - X_1}$$

$$Y_{i+1} = Y_i + \frac{Y_2 - Y_1}{X_2 - X_1}$$

Entre dos puntos la ordenada es aumentada en $\frac{Y_2 - Y_1}{X_2 - X_1}$ que es inferior a uno, según nuestra hipótesis. Si despreciamos este término no aumentando Y, cometemos un error de $\frac{Y_2 - Y_1}{X_2 - X_1}$. Así, en el siguiente punto tendremos:

$$Y_i + 2 = Y_i - 2 * \frac{Y_2 - Y_1}{X_2 - X_1}$$

Si no aumentamos Y durante n puntos, el error acumulado será igual a $n * \frac{Y_2 - Y_1}{X_2 - Y_1}$. Cuando este valor sea superior a Ø,5, será necesario incrementar la ordenada. El procedimiento utilizado en el programa se deriva de lo que acabamos de decir. Se controla una

variable llamada ERR que no contiene el error propiamente dicho, que es un número fraccionario, sino un número entero igual a:

$n * (Y_2 - Y_1) + \frac{X_2 - X_1}{2}$. El error será superior a ∅,5 si este número sobrepasa $X_2 - X_1$.

$$n * (Y_2 - Y_1) + \frac{X_2 - X_1}{2} > X_2 - X_1 = > n * \frac{(Y_2 - Y_1)}{X_2 - X_1} > ∅,5$$

Entonces habrá que incrementar la ordenada y sustraer el valor $X_2 - X_1$ a la variable ERR, que viene a ser lo mismo que sustraer 1 del error. De esta forma se obtiene el trazado completo de la recta haciendo variar X_i desde X_1 a X_2.

El programa general tiene una fase de inicialización destinada a calcular $| X_2 - X_1 |$ e $| Y_2 - Y_1 |$ y a determinar el incremento que hay que añadir a X y a Y (− 1 o + 1) según el signo de $X_2 - X_1$ y de $Y_2 - Y_1$. Encontramos a continuación dos bucles para el trazado de la recta correspondiente a uno de los casos: $| X_2 - X_1 | > | Y_2 - Y_1 |$ y $| X_2 - X_1 | < | Y_2 - Y_1 |$.

Entonces es posible trazar cualquier recta mientras sus coordenadas iniciales y finales estén comprendidas entre los límites admitidos y que sean diferentes. Al igual que para las funciones gráficas, antes de trazar la recta se procederá a posicionar los atributos gráficos.

```
                 00010 ;*************************************************************************
                 00020 ;
                 00030 ;  TRAZADO DE UNA RECTA EN LA PANTALLA
                 00040 ;
                 00050 ;  ENTRADA: D CONTIENE LA ORDENADA INICIAL Y1
                 00060 ;           E CONTIENE LA ABSCISA INICIAL X1
                 00070 ;           H CONTIENE LA ORDENADA FINAL Y2
                 00080 ;           L CONTIENE LA ABSCISA FINAL X2
                 00090 ;
                 00100 ;*************************************************************************
                 00110 ;
0000 0601        00120 RECTA LD   B,1      ; EL INCREMENTO EN Y VALE 1 POR DEFECTO
0002 48          00130       LD   C,B      ; EL INCREMENTO EN X VALE 1 POR DEFECTO
0003 7A          00140       LD   A,D      ; A = Y2
0004 94          00150       SUB  H        ; A = Y2 - 1
0005 3004        00160       JR   NC,GO    ; SI Y2 - Y1 > = 0 NO HACER NADA
0007 06FF        00170       LD   B,0FFH   ; SI NO TOMAR - 1 COMO INCREMENTO
0009 7C          00180       LD   A,H      ;


000A 92          00190       SUB  D        ; A = -(Y2 - Y1)
000B 57          00200 GO    LD   D,A      ; D = ABS(Y2 - Y1)
000C 7B          00210       LD   A,E      ; A = X2
000D 95          00220       SUB  L        ; A = X2 - X1
000E 3004        00230       JR   NC,G1    ; SI X2 - X1 > = 0 NO HACER NADA
0010 0EFF        00240       LD   C,0FFH   ; SI NO TOMAR - 1 COMO INCREMENTO
0012 7D          00250       LD   A,L      ;
0013 93          00260       SUB  E        ; A = - (X2 - X1)
0014 5F          00270 G1    LD   E,A      ; E = ABS (X2 - X1)
0015 ED436100    00280       LD   (G3),BC  ; RESGUARDAR LOS INCREMENTOS EN G3
0019 7A          00290       LD   A,D      ; A = ABS (Y2 - Y1)
```

```
001A BB       00300         CP    E           ; ¿ES ABS (Y2 - Y1) > = ABS (X2 - X1)?
001B 3021     00310         JR    NC,G2       ; SI ES SI PASAR AL SEGUNDO BUCLE
001D 43       00320         LD    B,E         ; B = ABS (X2 - X1): CONTADOR DE BUCLE
001E 4B       00330         LD    C,E         ; C = ERR
001F CB39     00340         SRL   C           ; ERR = ABS (X2 - X1)/2
0021 CD6300   00350 G5      CALL  SET         ; VISUALIZACION DEL SEGMENTO H,L
0024 3A6100   00360         LD    A,(G3)      ; INCREMENTO EN X
0027 85       00370         ADD   A,L         ; AÑADIR EL INCREMENTO A XI
0028 6F       00380         LD    L,A         ; NUEVO XI
0029 79       00390         LD    A,C         ; A = ERR
002A 82       00400         ADD   A,D         ; A = ERR + ABS (Y2 - Y1)
002B 4F       00410         LD    C,A         ; ERR = ERR + ABS (Y2 - Y1)
002C 3805     00420         JR    C,ZC        ; ERR> = 256
002E 93       00430         SUB   E           ; SUSTRAER ABS (X2 - X1)
002F 3809     00440         JR    C,G4        ; SI ERR < ABS (X2 - X1) PASAR AL PUNTO
0031 1801     00450         JR    ZD          ; SIGUIENTE
0033 93       00460 ZC      SUB   E           ; SUSTRAER ABS (X2 - X1)
0034 4F       00470 ZD      LD    C,A         ; ERR = ERR - ABS (X2 - X1)
0035 3A6200   00480         LD    A,(GE + 1)  ; INCREMENTO EN Y
0038 84       00490         ADD   A,H         ; INCREMENTAR Y
0039 67       00500         LD    H,A         ; NUEVA Y
003A 10E5     00510 G4      DJNZ  G5          ; BUCLE PARA VISULIZAR TODOS LOS
              00520                           ; PUNTOS
003C 181F     00530         JR    G6          ; VISUALIZAR EL PUNTO FINAL
003E 42       00540 G2      LD    B,D         ; B = ABS (Y2 - Y1): CONTADOR DE BUCLE
003F 4A       00550         LD    C,D         ; ERR = C
0040 CB39     00560         SRL   C           ; ERR = ABS (Y2 - Y1)/2
0042 CD6300   00570 G7      CALL  SET         ; VISUALIZACION DE UN SEGMENTO
0045 3A6200   00580         LD    A,(GE + 1)  ; INCREMENTO EN Y
0048 84       00590         ADD   A,H         ; INCREMENTAR Y
0049 67       00600         LD    H,A         ; NUEVA Y
004A 79       00610         LD    A,C         ; A = ERR
004B 83       00620         ADD   A,E         ; A = ERR + ABS (X2 - X1)
004C 4F       00630         LD    C,A         ; ERR = A
004D 3805     00640         JR    C,ZE        ; ERR > = 256
004F 92       00650         SUB   D           ; SUSTRAER ABS (Y2 - Y1)
0050 3909     00660         JR    C,G8        ; SI ERR < ABS (Y2 - Y1) PASAR AL PUNTO
0052 1801     00670         JR    ZH          ; SIGUIENTE
0054 92       00680 ZE      SUB   D           ; SUSTRAER ABS (Y2 - Y1)
0055 4F       00690 ZH      LD    C,A         ; ERR = ERR - ABS (Y2 - Y1)
0056 3A6100   00700         LD    A,(G3)      ; INCREMENTO EN X
0059 85       00710         ADD   A,L         ; INCREMENTAR X
005A 6F       00720         LD    L,A         ; NUEVA X
005B 10E5     00730 G8      DJNZ  G7          ; BUCLE PARA VISUALIZAR TODOS
              00740                           ; LOS PUNTOS
005D CD6300   00750 G6      CALL  SET         ; VISUALIZACION DEL ULTIMO PUNTO
0060 C9       00760         RET               ;
0002          00770 G3      DEFS  2           ;
0063          00780 SET     EQU   $           ; EL SUBPROGRAMA SET DEBERA
              00790                           ; IMPLANTARSE AQUI
0000          00800         END
```

4.1.5. Inserción de un mensaje en la pantalla: MES

Los dibujos representativos de los caracteres ASCII utilizados por el Basic se guardan dentro de una tabla en memoria muerta. Esta tabla se sitúa entre las direcciones 3DØØH y 3FFFH, ambas inclusive. Contiene todos los códigos ASCII a partir del código de número 32 (espacio), hasta el código de número 127 (©), en el orden de los códigos crecientes. Cada dibujo es memorizado por los ocho octetos que constituyen el carácter.

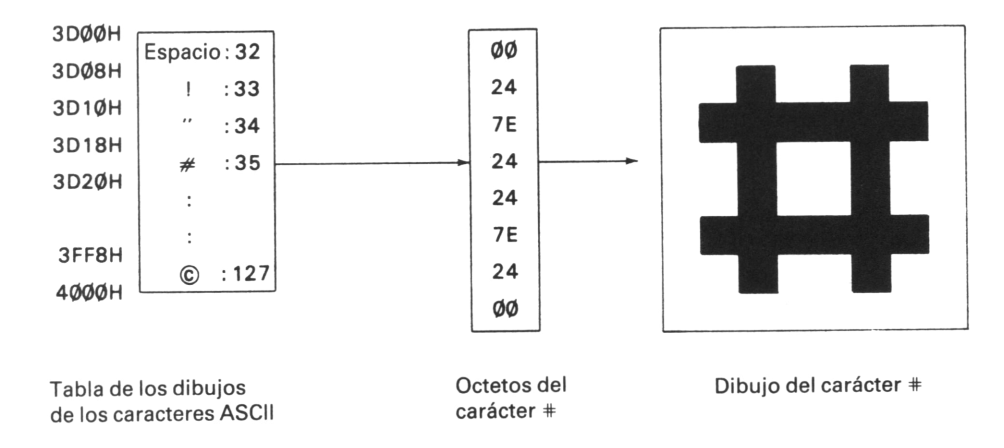


Tabla de los dibujos de los caracteres ASCII

Octetos del carácter #

Dibujo del carácter #

El subprograma MES utiliza esta tabla para dibujar los caracteres ASCII en la pantalla. MES utiliza el octeto almacenado en la variable ATRB como atributo para el dibujo de los caracteres de un mensaje. Este se da bajo la forma de una serie de caracteres ASCII (obtenida, por ejemplo, con la orden DEFM), pudiendo contener algunos caracteres especiales (obtenidos mediante las órdenes DEFB o DEFW). El subprograma MES reconoce cuatro tipos de caracteres especiales que permiten actuar sobre el desarrollo de la visualización del mensaje:

Carácter especial	*Significado*
Ø	Este carácter indica el fin del mensaje.
13	Provoca un salto de línea (retorno de carro).
2 seguido de un atributo	Coloca el octeto siguiente del mensaje en la variable ATRB que contiene el atributo para el dibujo de los caracteres.
127 + n	Salto de n espacios (1 ≤ n ≤ 128).

He aquí un ejemplo de mensaje que puede ser utilizado por MES:

```
0000 BC         00100 MESSA  DEFB   140        ; DECALAJE PARA CENTRADO
0001 02         00110        DEFB   2,31H      ; AZUL SOBRE FONDO AMARILLO
     31
0003 42         00120        DEFM   'Bonjour'
     6F 6E 6A   6F 75 72
000A 0D         00130        DEFB   13,13      ; SALTOS DE LINEA
     0D
000C 02         00140        DEFB   2,0B8H     ; NEGRO SOBRE BLANCO Y
     B8                                          PARPADEO
000E 4A         00150        DEFM   'Je suis le ZX SPECTRUM'
     65 20 73   75 69 73 20 6C
     65 20 5A   58 20 53 50 45
     43 54 52   55 4D
0024 00         00160        DEFB   0          ; FIN DEL MENSAJE
```

Listado del programa MES:

```
          00010  ; *****************************************************************************
          00020  ;
          00030  ;  SUBPROGRAMA DE VISUALIZACION DE UN MENSAJE EN LA PANTALLA
          00040  ;
          00050  ;  ENTRADA: D   = NUMERO DE LINEA DONDE COMIENZA LA VISUALIZACION
          00060  ;           E   = NUMERO DE COLUMNA DONDE COMIENZA LA VISUALIZACION
          00070  ;           HL  = DIRECCION DEL PRIMER OCTETO DEL MENSAJE
          00080  ;
          00090  ; *****************************************************************************
          00100  ;
0000 7E        00110 MES  LD    A,(HL)     ; CARACTER EN A
0001 23        00120      INC   HL         ; INCREMENTAR PUNTERO
0002 B7        00130      OR    A          ; ¿COMPROBAR SI A = 0?
0003 C8        00140      RET   Z          ; SI ES SI RETORNO; FIN DEL MENSAJE
0004 FE0D      00150      CP    0DH        ; ¿COMPROBAR A = RETORNO DE CARRO?
0006 2005      00160      JR    NZ,ZI      ; NO
0008 14        00170      INC   D          ; SALTAR UNA LINEA
0009 1E00      00180      LD    E,0        ; COLOCARSE DE NUEVO AL INICIO DE LA LINEA
000B 18F3      00190      JR    MES        ; PASAR AL CARACTER SIGUIENTE
000D CB7F      00200 ZI   BIT   7,A        ; ¿COMPROBAR SI A > = 128?
000F 2811      00210      JR    Z,ZJ       ; NO
0011 D67F      00220      SUB   127        ; A CONTIENE EL NUMERO DE ESPACIOS
0013 83        00230      ADD   A,E        ; AÑADIR AL NUMERO DE COLUMNAS
0014 4F        00240      LD    C,A        ; GUARDAR RESULTADO EN C
0015 E61F      00250      AND   1FH        ; MASCARA PARA RECUPERAR EL NUMERO
0017 5F        00260      LD    E,A        ; DE COLUMNAS
0018 79        00270      LD    A,C        ; RECUPERAR RESULTADO
0019 E6E0      00280      AND   0E0H       ; NUMERO DE LINEAS * 32
001B 07        00290      RLCA             ; DIVIDIR POR 32 PARA OBTENER
001C 07        00300      RLCA             ; EL NUMERO DE LINEAS
001D 07        00310      RLCA             ;
001E 82        00320      ADD   A,D        ; AÑADIR RESULTADO A D
001F 57        00330      LD    D,A        ; NUEVO NUMERO DE LA LINEA
0020 18DE      00340      JR    MES        ; PASAR AL SIGUIENTE CARACTER
0022 FE02      00350 ZJ   CP    2          ; ¿COMPROBAR SI A = 2?
0024 2007      00360      JR    NZ,ZK      ; NO
0026 7E        00370      LD    A,(HL)     ; LECTURA DE ATRIBUTO
0027 23        00380      INC   HL         ; INCREMENTAR PUNTERO
0028 328600    00390      LD    (ATRB),A   ; COLOCAR EL ATRIBUTO EN ATRB
002B 18D3      00400      JR    MES        ; PASAR AL SIGUIENTE CARACTER
002D CD3A00    00410 ZK   CALL  AFI        ; VISUALIZACION DEL CARACTER ASCII
0030 1C        00420      INC   E          ; COLUMNA SIGUIENTE
0031 CB6B      00430      BIT   5,E        ; ¿COLUMNA > = 32?
0033 28CB      00440      JR    Z,MES      ; NO
0035 14        00450      INC   D          ; PASAR A LA LINEA SIGUIENTE
0036 1E00      00460      LD    E,0        ; COLOCARSE AL INICIO DE LA LINEA
0038 18C6      00470      JR    MES        ; PASAR AL SIGUIENTE CARACTER
          00480  ; *****************************************************************************
          00490  ;
          00500  ;  SUBPROGRAMA DE VISUALIZACION DE UN CARACTER ASCII
          00510  ;
          00520  ;  ENTRADA: D = NUMERO DE LINEA
          00530  ;           E = NUMERO DE COLUMNA
          00540  ;           A = CODIGO ASCII DEL CARACTER
          00550  ;
          00560  ;  NO SE MODIFICA NINGUN REGISTRO SALVO AF
          00570  ;
          00580  ; *****************************************************************************
          00590  ;
003A C5        00600 AFI  PUSH  BC         ; GUARDAR REGISTROS
003B E5        00610      PUSH  HL         ;
003C D5        00620      PUSH  DE         ;
003D F5        00630      PUSH  AF         ;
003E CD5F00    00640      CALL  TRAN       ; BUSQUEDA DIRECCION DEL PUNTO
0041 F1        00650      POP   AF         ; CARACTER ASCII
0042 2600      00660      LD    H,0        ;
0044 6F        00670      LD    L,A        ; CODIGO CARACTER EN HL
0045 29        00680      ADD   HL,HL      ; CODIGO * 2
0046 29        00690      ADD   HL,HL      ; CODIGO * 4
0047 29        00700      ADD   HL,HL      ; CODIGO * 8
0048 01003C    00710      LD    BC,3C00H   ; INICIO ZONA DE DIBUJO DE LOS CARACTERES -256
004B 09        00720      ADD   HL,BC      ; DIRECCION DEL DIBUJO DEL CARACTER
004C 0608      00730      LD    B,8        ; 8 OCTETOS EN EL CARACTER
```

```
004E 7E       00740 ZM    LD    A,(HL)    ; OCTETO DE LA TABLA
004F 12       00750       LD    (DE),A    ; TRANSFERENCIA A LA MEMORIA VIDEO
0050 23       00760       INC   HL        ; OCTETO SIGUIENTE DE LA TABLA
0051 14       00770       INC   D         ; DIRECCION VIDEO SIGUIENTE
0052 10FA     00780       DJNZ  ZM        ; BUCLE PARA TRANSFERIR LOS 8 OCTETOS
0054 D1       00790       POP   DE        ; POSICION EN LA PANTALLA
0055 CD7800   00800       CALL  TATR      ; BUSQUEDA DE LA DIRECCION DEL OCTETO DE
0058 3A8600   00810       LD    A,(ATRB)  ATRIBUTOS
005B 77       00820       LD    (HL),A    ; ATRIBUTO A UTILIZAR
005C E1       00830       POP   HL        ; ASIGNAR ESTE ATRIBUTO AL CARACTER
005D C1       00840       POP   BC        ; RESTAURAR REGISTROS
005E C9       00850       RET             ;
                                          ;

              00860 ; ******************************************************************************
              00870 ;
              00880 ;  SUBPROGRAMA DE CONVERSION
              00890 ;
              00900 ;  ENTRADA: D = NUMERO DE LINEA
              00910 ;           E = NUMERO DE COLUMNA
              00920 ;
              00930 ;  SALIDA: DE = DIRECCION DEL PRIMER OCTETO DEL CARACTER
              00940 ;               EN MEMORIA VIDEO DE LOS PUNTOS
              00950 ;
              00960 ; ******************************************************************************
              00970 ;
005F 7A       00980 TRAN  LD    A,D       ; NUMERO DE COLUMNA
0060 1640     00990       LD    D,40H     ; INICIO PRIMERA ZONA
0062 D608     01000       SUB   8         ; QUITAR LONGITUD DE LA ZONA
0064 380A     01010       JR    C,ZL      ; SI PRIMERA ZONA
0066 1648     01020       LD    D,48H     ; INICIO SEGUNDA ZONA
0068 D608     01030       SUB   8         ; QUITAR LONGITUD DE LA ZONA
006A 3804     01040       JR    C,ZL      ; SI SEGUNDA ZONA
006C 1650     01050       LD    D,50H     ; ULTIMA ZONA
006E D608     01060       SUB   8         ; QUITAR LONGITUD DE LA ZONA
0070 C608     01070 ZL    ADD   A,8       ; NUMERO DE LINEA EN LA ZONA
0072 0F       01080       RRCA            ; MULTIPLICACION DEL NUMERO DE
0073 0F       01090       RRCA            ; LINEA POR 32
0074 0F       01100       RRCA            ;
0075 83       01110       ADD   A,E       ; AÑADIR A COLUMNA PARA OBTENER
0076 5F       01120       LD    E,A       ; EL OCTETO DE MENOS PESO DE
0077 C9       01130       RET             ; LA DIRECCION VIDEO
              01140 ; ******************************************************************************
              01150 ;
              01160 ;  SUBPROGRAMA DE CONVERSION
              01170 ;
              01180 ;  ENTRADA: D = NUMERO DE LINEA
              01190 ;           E = NUMERO DE COLUMNA
              01200 ;
              01210 ;  SALIDA: HL = DIRECCION DEL OCTETO DE ATRIBUTOS ASOCIADO
              01220 ;
              01230 ; ******************************************************************************
              01240 ;
0078 2600     01250 TATR  LD    H,0       ;
007A 6A       01260       LD    L,D       ; HL CONTIENE EL NUMERO DE LINEA
007B 29       01270       ADD   HL,HL     ; NUMERO DE LINEA * 2
007C 29       01280       ADD   HL,HL     ; NUMERO DE LINEA * 4
007D 29       01290       ADD   HL,HL     ; NUMERO DE LINEA * 8
007E 29       01300       ADD   HL,HL     ; NUMERO DE LINEA * 16
007F 29       01310       ADD   HL,HL     ; NUMERO DE LINEA * 32
0080 D5       01320       PUSH  DE        ; GUARDAR POSICION
0081 1658     01330       LD    D,58H     ; INICIO ZONA DE ATRIBUTOS
0083 19       01340       ADD   HL,DE     ; HL = DIRECCION OCTETO DE ATRIBUTOS
0084 D1       01350       POP   DE        ; RESTAURAR POSICION
0085 C9       01360       RET             ;
0001          01370 ATRB  DEFS  1         ; ATRIBUTO A UTILIZAR
0000          01380       END
```

4.1.6. Visualización de caracteres gráficos

Es fácil modificar el programa MES precedente para que permita la visualización de caracteres gráficos. Para ello hagamos que el carácter especial 1 solicite la visualización del carácter gráfico cuyos ocho caracteres se encuentran después del carácter especial 1.

El mensaje gráfico siguiente podrá ser utilizado con el subprograma MES modificado:

```
MESA    DEFB    2, 2              ; Rojo sobre fondo negro
        DEFB    1                 ; Carácter gráfico
        DEFB    14H, 1CH, 1CH, 14H, 5DH, 7EH, 5DH, 77H
        DEFB    13                ; Salto de línea
        DEFM    'Bonjour'
        DEFB    Ø
```

He aquí el listado del subprograma MES modificado al que habrá que añadir los tres subprogramas (AFI, TRAN, TATR) que MES utiliza.

```
              00010  ; ******************************************************************************
              00020  ;
              00030  ;  SUBPROGRAMA DE VISUALIZACION DE UN MENSAJE EN LA PANTALLA
              00040  ;
              00050  ;  ENTRADA: D  = NUMERO DE LINEA DONDE COMIENZA LA VISUALIZACION
              00060  ;           E  = NUMERO DE COLUMNA DONDE COMIENZA LA VISUALIZACION
              00070  ;           HL = DIRECCION DEL PRIMER OCTETO DEL MENSAJE
              00080  ;
              00090  ; ******************************************************************************
              00100  ;
0000 7E       00110 MES    LD     A,(HL)    ; CARACTER EN A
0001 23       00120        INC    HL        ; INCREMENTAR PUNTERO
0002 B7       00130        OR     A         ; ¿COMPROBAR SI A = Ø?
0003 C8       00140        RET    Z         ; SI ES SI RETORNO; FIN DEL MENSAJE
0004 FE0D     00150        CP     0DH       ; ¿COMPROBAR SI A = RETORNO DE CARRO?
0006 2005     00160        JR     NZ,ZI     ; NO
0008 14       00170        INC    D         ; SALTAR UNA LINEA
0009 1E00     00180        LD     E,0       ; COLOCARSE DE NUEVO EN INICIO DE LINEA
000B 18F3     00190        JR     MES       ; PASAR AL CARACTER SIGUIENTE
000D CB7F     00200 ZI     BIT    7,A       ; ¿COMPROBAR SI A > = 128?
000F 2811     00210        JR     Z,ZJ      ; NO
0011 D67F     00220        SUB    127       ; A CONTIENE EL NUMERO DE ESPACIOS
0013 83       00230        ADD    A,E       ; AÑADIR AL NUMERO DE COLUMNAS
0014 4F       00240        LD     C,A       ; GUARDAR RESULTADO EN C
0015 E61F     00250        AND    1FH       ; MASCARA PARA RECUPERAR EL NUMERO
0017 5F       00260        LD     E,A       ; DE COLUMNAS
0018 79       00270        LD     A,C       ; RECUPERAR RESULTADO
0019 E6E0     00280        AND    0E0H      ; NUMERO DE LINEAS * 32
001B 07       00290        RLCA             ; DIVIDIR POR 32 PARA OBTENER
001C 07       00300        RLCA             ; EL NUMERO DE LINEAS
001D 07       00310        RLCA             ;
001E 82       00320        ADD    A,D       ; AÑADIR RESULTADO A D
001F 57       00330        LD     D,A       ; NUEVO NUMERO DE LINEA
0020 18DE     00340        JR     MES       ; PASAR AL CARACTER SIGUIENTE
0022 FE01     00350 ZJ     CP     1         ; ¿COMPROBAR SI A = 1?
0024 2018     00360        JR     NZ,ZP     ; NO
0026 D5       00370        PUSH   DE        ; GUARDAR POSICION
0027 CD7B00   00380        CALL   TRAN      ; BUSQUEDA DE LA DIRECCION DE LA MEMORIA VIDEO
002A 0608     00390        LD     B,8       ; 8 OCTETOS POR CARACTER
002C 7E       00400 ZQ     LD     A,(HL)    ; OCTETO DEL CARACTER
002D 12       00410        LD     (DE),A    ; VISUALIZACION
002E 23       00420        INC    HL        ; INCREMENTAR PUNTERO
002F 14       00430        INC    D         ; OCTETO SIGUIENTE DEL CARACTER
```

```
0030 10FA   00440       DJNZ  ZQ        ; BUCLE PARA VISUALIZAR LOS 8 OCTETOS
0032 D1     00450       POP   DE        ; RESTAURAR POSICION
0033 E5     00460       PUSH  HL        ; GUARDAR PUNTERO
0034 CD9400 00470       CALL  TATR      ; BUSQUEDA DIRECCION DEL OCTETO DE ATRIBUTOS
0037 3AA200 00480       LD    A,(ATRB); ATRIBUTO A UTILIZAR
003A 77     00490       LD    (HL),A    ; ASIGNAR ESTE ATRIBUTO AL CARACTER
003B E1     00500       POP   HL        ; RESTAURAR PUNTERO
003C 180E   00510       JR    ZR        ; PASAR AL CARACTER SIGUIENTE
003E FE02   00520       CP    2         ; ¿COMPROBAR SI A = 2?
0040 2007   00530       JR    NZ,ZK     ; NO
0042 7E     00540       LD    A,(HL)    ; LECTURA ATRIBUTO
0043 23     00550       INC   HL        ; INCREMENTAR PUNTERO
0044 32A200 00560       LD    (ATRB),A; COLOCAR EL ATRIBUTO EN ATRB
0047 18B7   00570       JR    MES       ; PASAR AL CARACTER SIGUIENTE
0049 CD5600 00580 ZK    CALL  AFI       ; VISUALIZACION DEL CARACTER ASCII
004C 1C     00590 ZR    INC   E         ; COLUMNA SIGUIENTE
004D CB6B   00600       BIT   5,E       ; ¿COLUMNA > = 32 ?
004F 28AF   00610       JR    Z,MES     ; NO
0051 14     00620       INC   D         ; PASAR A LA LINEA SIGUIENTE
0052 1E00   00630       LD    E,0       ; COLOCARSE EN INICIO DE LINEA
0054 18AA   00640       JR    MES       ; PASAR AL CARACTER SIGUIENTE
```

4.2. La impresora

La impresora del SPECTRUM tiene una resolución idéntica a la de la pantalla, lo que permite imprimir tanto textos (caracteres ASCII) como dibujos (caracteres gráficos). Está concebida para visualizar de una sola vez una línea de 32 caracteres. Así, el interpretador Basic controla un *buffer* de impresora de 256 octetos destinado a contener los 32 caracteres que hay que enviar hacia la impresora. Este buffer se sitúa entre las direcciones 5BØØH y 5BFFH. Los 32 primeros octetos del buffer contendrán los primeros 32 caracteres, los 32 octetos siguientes contendrán los segundos octetos de los 32 caracteres y así sucesivamente hasta los octavos octetos. El interpretador Basic controla una variable situada en la dirección 2368Ø que contiene el número de caracteres presentes en el buffer. Cuando el buffer está lleno, o cuando el carácter «retorno del carro» (código 13) es enviado a la impresora, el interpretador Basic envía el contenido del buffer hacia la impresora para sacarlo sobre papel, gracias al subprograma situado en la dirección ECDH. Si la impresora no está ligada al SPECTRUM, este subprograma no tiene ningún efecto.

El subprograma PRINT que indicamos seguidamente utiliza el subprograma situado en la dirección ECDH y la variable situada en la dirección 2368Ø para visualizar un mensaje constituido por caracteres ASCII, por retornos de carro y terminando por un octeto nulo.

Ejemplo de mensaje:

```
MESSA   DEFM    «Hola»
        DEFB    13
        DEFM    «Yo soy el SPECTRUM»
        DEFB    13,Ø
```

Dado que la impresión sólo es ordenada si el buffer está lleno o si se envía un retorno de carro, deberá terminarse el mensaje por un retorno de carro, salvo si su medida es múltiplo de 32.

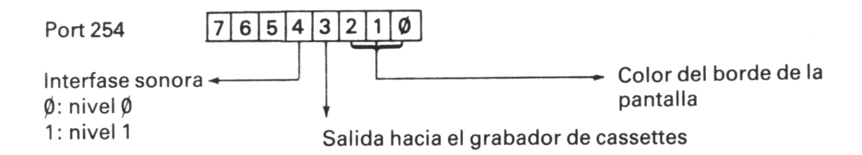
```
0ECD          00010 IMP    EQU   0ECDH   ; SUBPROGRAMA DE IMPRESION DEL
              00020                      ; BUFFER DE IMPRESORA
5C80          00030 NCAR   EQU   23680   ; NUMERO DE CARACTERES EN EL BUFFER
              00040  ; ************************************************************************
              00050  ;
              00060  ;  SUBPROGRAMA DE VISUALIZACION DE UN MENSAJE EN LA IMPRESORA
              00070  ;
              00080  ;  ENTRADA: HL = DIRECCION DEL PRIMER OCTETO DEL MENSAJE
              00090  ;
              00100  ; ************************************************************************
              00110  ;
0000 7E       00120 PRINT  LD    A,(HL)  ; CARACTER EN A
0001 23       00130        INC   HL      ; INCREMENTAR PUNTERO
0002 B7       00140        OR    A       ; ¿COMPROBAR SI A = Ø?
0003 C8       00150        RET   Z       ; SI ES SI RETORNO; FIN DEL MENSAJE
0004 E5       00160        PUSH  HL      ; GUARDAR PUNTERO
0005 FE0D     00170        CP    0DH     ; ¿COMPROBAR SI A = RETORNO DE CARRO?
0007 2824     00180        JR    Z,ZS    ; SI
0009 2600     00190        LD    H,0     ;
000B 6F       00200        LD    L,A     ; CODIGO DEL CARACTER EN HL
000C 29       00210        ADD   HL,HL   ; CODIGO * 2
000D 29       00220        ADD   HL,HL   ; CODIGO * 4
000E 29       00230        ADD   HL,HL   ; CODIGO * 8
000F 01003C   00240        LD    BC,3C00H; INICIO ZONA DEL DIBUJO DE LOS CARACTERES -256
0012 09       00250        ADD   HL,BC   ; DIRECCION DEL DIBUJO DEL CARACTER
0013 3A805C   00260        LD    A,(NCAR); PUNTERO SOBRE EL BUFFER DE IMPRESORA
0016 5F       00270        LD    E,A     ;
0017 165B     00280        LD    D,5BH   ; DE = PUNTERO SOBRE EL BUFFER DE IMPRESORA
0019 0608     00290        LD    B,8     ; 8 OCTETOS EN EL CARACTER
001B 7E       00300 ZT     LD    A,(HL)  ; OCTETO DE LA TABLA
001C 12       00310        LD    (DE),A  ; TRANSFERIDO A LA MEMORIA VIDEO
001D 23       00320        INC   HL      ; OCTETO SIGUIENTE DE LA TABLA
001E 7B       00330        LD    A,E     ; PASAR AL OCTETO SIGUIENTE DEL
001F C620     00340        ADD   A,32    ; CARACTER EN EL BUFFER
0021 5F       00350        LD    E,A     ; DE IMPRESORA
0022 10F7     00360        DJNZ  ZT      ; BUCLE PARA TRANSFERIR LOS 8 OCTETOS
0024 3A805C   00370        LD    A,(NCAR);
0027 3C       00380        INC   A       ; INCREMENTAR NUMERO DE CARACTERES
0028 32805C   00390        LD    (NCAR),A; EN EL BUFFER DE IMPRESORA
002B FE20     00400        CP    32      ; ¿FIN DE LINEA?
002D CCCD0E   00410 ZS     CALL  Z,IMP   ; SI ES SI IMPRESION DE LINEA
0030 E1       00420        POP   HL      ; RESTAURAR PUNTERO
0031 18CD     00430        JR    PRINT   ; PASAR AL CARACTER SIGUIENTE
0000          00440        END
```

Del mismo modo que para el subprograma MES, PRINT podrá ser modificado para permitir la impresión de caracteres gráficos.

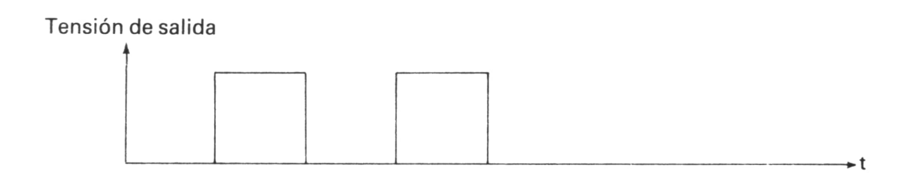
4.3. La interfase sonora

La interfase sonora del ZX SPECTRUM consta de una salida programable que puede tomar dos niveles diferentes y que está ligada a un altavoz a través de un pequeño amplificador.

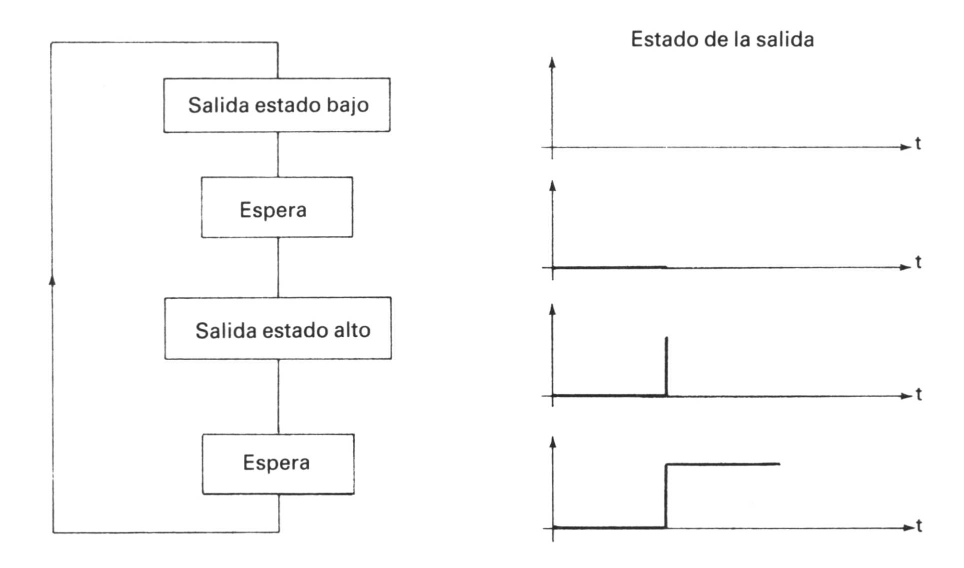
La programación de esta salida se realiza mediante el bit 4 del port 254. Este port sirve igualmente para la programación del borde de la pantalla y para la escritura de informaciones en el cassette.



Haciendo variar rápidamente la tensión tomada por esta salida, se genera una señal eléctrica que producirá un sonido al estar ligada a un amplificador con un altavoz. De este modo es posible crear una señal cuadrada cuya frecuencia podrá ajustarse con el fin de producir la nota musical deseada. Esta señal se representa por medio del esquema siguiente:



Para generar esta señal se utilizará un programa que responda al diagrama del flujo siguiente:



Para que el sonido no dure eternamente como en este diagrama de flujo, tendrá que utilizarse un segundo contador que limite el número de pulsos generados. Así será posible modificar la duración de la nota, escogiendo el número de pulsos enviados, y modificar su tono, disminuyendo o aumentando la duración creada por el bucle de espera. Cuanto más grande sea el bucle de espera más grave será el sonido, y cuanto más pequeño, más agudo.

```
                 00010  ; ***************************************************************************
                 00020  ;
                 00030  ;  EL PROGRAMA SIGUIENTE GENERA UNA NOTA DE DURACION
                 00040  ;  LIMITADA Y PARA EL MICROPROCESADOR (INSTRUCCION HALT)
                 00050  ;
                 00060  ; ***************************************************************************
                 00070  ;
8000             00080         ORG    8000H
0007             00090 COLOR   EQU    7         ; COLOR BLANCO PARA EL BORDE
8000 F3          00100 INICIO  DI               ; PROHIBIR INTERRUPCIONES PARA
                 00110                          ; TENER UN SONIDO PURO
8001 3E07        00120         LD     A,COLOR   ; COLOR DEL BORDE EN A
8003 0E00        00130         LD     C,0       ; CONTADOR DE DURACION DE LA NOTA
8005 D3FE        00140 LOOP    OUT    (254),A   ; MODIFICACION DE LA SALIDA
8007 06C8        00150         LD     B,200     ; CONTADOR DE BUCLE DE ESPERA
8009 10FE        00160 WAIT    DJNZ   WAIT      ; ESPERA
800B EE10        00170         XOR    10H       ; EL BIT 4 DE A ES COMPLEMENTADO
800D 0D          00180         DEC    C         ; DECREMENTAR CONTADOR DE DURACION
800E 20F5        00190         JR     NZ,LOOP   ; SI EL CONTADOR NO ES NULO CONTINUAR
8010 76          00200         HALT             ; PARO DEL Z 80
8000             00210         END    INICIO
```

En el programa de ensamblador siguiente es posible evaluar la frecuencia de la nota generada calculando la duración producida por el bucle de espera. Entre dos valores de la tensión de salida, el microprocesador utiliza un número de ciclos de reloj igual a $11 + 7 + 149 * 13 + 8 + 7 + 12 = 1986$. Si la frecuencia del reloj del ZX SPECTRUM está fijada a 3,25 MHz, la del sonido generado será 3,25 MHz/1986, o sea 1636 Hz (frecuencia audible).

Las notas musicales tienen una frecuencia determinada. También es posible calcular el número de inicialización del contador de bucle de espera para producir la nota deseada. A este número le llamaremos N. La frecuencia generada viene dada por la fórmula siguiente:

$$F = \frac{3{,}25\ 10^{6}}{46 + (N - 1) * 3} \quad \text{de donde} \quad N = \frac{1}{13}\left(\frac{3{,}25\ 10^{6}}{F} - 46\right) + 1$$

Dado que la duración del bucle de espera es variable, según la nota emitida deberá ajustarse el contador de duración para generar notas de la misma duración. Sean D_1 o D_2 los valores que inicializan a los contadores de bucle de duración respectivamente para las frecuencias F_1 y F_2. Tenemos la siguiente fórmula:

$$\frac{D_1}{F_1} = \frac{D_2}{F_2}$$

Una vez escogida la duración de una nota, esta fórmula permitirá determinar todos los números D que sirven para inicializar los contadores de bucle. Los valores de estos parámetros correspondientes a la escala se dan en la siguiente tabla:

Nota	*DO*	*RE*	*MI*	*FA*	*SOL*	*LA*	*SI*	*DO*
Frecuencia (Hz)	1Ø47	1175	1319	1397	1568	176Ø	1976	2Ø93
Contador de bucle de espera	236	21Ø	187	176	157	14Ø	124	117
Contador de bucle de duración	128	143	161	17Ø	191	214	241	255

El siguiente programa recorre constantemente la escala en un sentido y luego en el otro. Con el fin de animar un juego podrá utilizarse este programa para generar una pequeña melodía al final de la partida, modificando los valores de las constantes reagrupadas en la tabla TAB.

```
              00010  ; ******************************************************************
              00020  ;
              00030  ;  EL SIGUIENTE PROGRAMA GENERA ETERNAMENTE LA
              00040  ;  ESCALA EN UN SENTIDO Y EN OTRO
              00050  ;
              00060  ; ******************************************************************
              00070  ;
8000          00080         ORG    8000H
0007          00090 COLOR   EQU    7         ; COLOR BLANCO PARA EL BORDE
8000 F3       00100 INICIO  DI               ; PROHIBIR INTERRUPCIONES PARA
              00110                          ; TENER UN SONIDO PURO
8001 213380   00120         LD     HL,TAB    ; INICIALIZACION DEL PUNTERO SOBRE TAB
              00130  ;
              00140  ;  RECORRER ESCALA AL DERECHO
              00150  ;
8004 1E08     00160 BUCLE   LD     E,8       ; 8 NOTAS EN LA ESCALA
8006 4E       00170 U       LD     C,(HL)    ; CONTADOR DE BUCLE DE ESPERA
8007 23       00180         INC    HL        ; INCREMENTAR PUNTERO
8008 56       00190         LD     D,(HL)    ; CONTADOR DE DURACION
8009 23       00200         INC    HL        ; INCREMENTAR PUNTERO
800A CD1E80   00210         CALL   SON       ; GENERACION DE UNA NOTA
800D 1D       00220         DEC    E         ; CONTADOR DE NUMERO DE NOTAS
800E 20F6     00230         JR     NZ,ZU     ; PASAR A LA NOTA SIGUIENTE
              00240  ;
              00250  ;  RECORRER LA ESCALA A LA INVERSA
              00260  ;
8010 1E08     00270         LD     E,8       ; 8 NOTAS EN LA ESCALA
8012 2B       00280 ZV      DEC    HL        ; DECREMENTAR EL PUNTERO
8013 56       00290         LD     D,(HL)    ; CONTADOR DE DURACION
8014 2B       00300         DEC    HL        ; DECREMENTAR PUNTERO
8015 4E       00310         LD     C,(HL)    ; CONTADOR DE BUCLE DE ESPERA
8016 CD1E80   00320         CALL   SON       ; GENERACION DE UNA NOTA
8019 1D       00330         DEC    E         ; CONTADOR DE NUMERO DE NOTAS
801A 20F6     00340         JR     NZ,ZV     ; PASAR A LA NOTA SIGUIENTE
801C 18E6     00350         JR     BUCLE     ; RECOMENZAR ETERNAMENTE
```

```
            00360  ; ***************************************************************************
            00370  ;
            00380  ;  SUBPROGRAMA DE SONIDO
            00390  ;
            00400  ;  ENTRADA: D = CONTADOR DE BUCLE DE DURACION
            00410  ;           C = CONTADOR DE BUCLE DE ESPERA
            00420  ;
            00430  ; ***************************************************************************
            00440  ;
801E 3E07   00450 SON    LD     A,COLOR  ; COLOR DEL BORDE EN A
8020 D3FE   00460 LOOP   OUT    (254),A  ; MODIFICACION DE LA SALIDA
8022 41     00470        LD     B,C      ; CONTADOR DE BUCLE DE ESPERA
8023 10FE   00480 WAIT   DJNZ   WAIT     ; ESPERA
8025 EE10   00490        XOR    10H      ; EL BIT 4 DE A ES COMPLEMENTADO
8027 15     00500        DEC    D        ; DECREMENTAR EL CONTADOR DE DURACION
8028 20F6   00510        JR     NZ,LOOP  ; SI EL CONTADOR NO ES NULO CONTINUAR
802A 010004 00520        LD     BC,400H  ; CONTADOR DE BUCLE DE ESPERA
802D 0B     00530 LA     DEC    BC       ; ESPERA ENTRE CADA NOTA
802E 78     00540        LD     A,B      ;
802F B1     00550        OR     C        ; ¿COMPROBAR SI BC = Ø?
8030 20FB   00560        JR     NZ,LA    ; SI BC < > Ø,BUCLE
8032C9      00570        RET             ; RETORNO
            00580  ;
            00590  ;  TABLA DE LAS NOTAS
            00600  ;
8033 EC     00610 TAB    DEFB   236      ; ESPERA DO
8034 80     00620        DEFB   128      ; DURACION DO
8035 D2     00630        DEFB   210      ; ESPERA RE
8036 8F     00640        DEFB   143      ; DURACION RE
8037 BB     00650        DEFB   187      ; ESPERA MI
8038 A1     00660        DEFB   161      ; DURACION MI
8039 B0     00670        DEFB   176      ; ESPERA FA
803A AA     00680        DEFB   170      ; DURACION FA
803B 9D     00690        DEFB   157      ; ESPERA SOL
803C BF     00700        DEFB   191      ; DURACION SOL
803D 8C     00710        DEFB   140      ; ESPERA LA
803E D6     00720        DEFB   214      ; DURACION LA
803F 7C     00730        DEFB   124      ; ESPERA SI
8040 F1     00740        DEFB   241      ; DURACION SI
8041 75     00750        DEFB   117      ; ESPERA DO
8042 FF     00760        DEFB   255      ; DURACION DO
8000        00770        END    INICIO
```

Para obtener sonoridades más complejas, la duración producida por el bucle de espera deberá modularse, modificando su duración a lo largo del tiempo. La modificación más simple consiste en decrementar la variable que inicializa el contador del bucle de espera cada vez que la salida cambie de estado. De esta manera se obtiene un sonido muy interesante que se utiliza en numerosos juegos.

```
            00010  ; ***************************************************************************
            00020  ;
            00030  ;  EL SIGUIENTE PROGRAMA GENERA UN SONIDO DE DURACION
            00040  ;  LIMITADA Y PARA EL MICROPROCESADOR (INSTRUCCION HALT)
            00050  ;
            00060  ; ***************************************************************************
            00070  ;
8000        00080        ORG    8000H
0007        00090 COLOR  EQU    7        ; COLOR BLANCO PARA EL BORDE
8000 F3     00100 INICIO DI              ; PROHIBIR INTERRUPCIONES PARA
            00110                        ; TENER UN SONIDO PURO
8001 3E07   00120        LD     A,COLOR  ; COLOR DEL BORDE EN A
8003 0E00   00130        LD     C,0      ; CONTADOR DE DURACION DE SONIDO
8005 D3FE   00140 LOOP   OUT    (254),A  ; MODIFICACION DE LA SALIDA
8007 41     00150        LD     B,C      ; CONTADOR DE BUCLE DE ESPERA
8008 10FE   00160 WAIT   DJNZ   WAIT     ; ESPERA
800A EE10   00170        XOR    10H      ; EL BIT 4 DE A ES COMPLEMENTADO
800C 0D     00180        DEC    C        ; DECREMENTAR CONTADOR DE DURACION
800D 20F6   00190        JR     NZ,LOOP  ; SI EL CONTADOR NO ES NULO CONTINUAR
800F 76     00200        HALT            ; PARO DEL Z 8Ø
8000        00210        END    INICIO
```

Modificando la duración del bucle de espera por procedimientos diferentes se obtendrán sonoridades más o menos buenas. Experiméntelas y retenga aquellas que le parezcan mejores. Combinando estas sonoridades elementales, a veces podrán obtenerse otras mejores.

4.4. El teclado

Ocho ports de entrada/salida permiten leer todas las teclas del teclado. Cada port permite leer una media hilera de cinco teclas tal como se indica en la tabla siguiente:

Número del port	*Media hilera*	*bit 0*	*bit 1*	*bit 2*	*bit 3*	*bit 4*
FEFEH	CAPS SHIFT à V	CAPS SHIFT	Z	X	C	V
FDFEH	A à G	A	S	D	F	G
FBFEH	Q à T	Q	W	E	R	T
F7FEH	1 à 5	1	2	3	4	5
EFFEH	Ø à 6	Ø	9	8	7	6
DFFEH	P à Y	P	O	I	U	Y
BFFEH	ENTER à H	ENTER	L	K	J	H
7FFEH	SPACE à B	SPACE	SYMBOL SHIFT	M	N	B

Una tecla está pulsada si el bit correspondiente del port es nulo. Si este bit vale 1, la tecla no está pulsada.

Contrariamente al port 254 utilizado para la interfase sonora, y para especificar el borde de la pantalla que era un port de ocho bits, los ports que se indican a continuación son ports de 16 bits. Será necesario, pues, especificar el octeto de más peso y el octeto de menos peso del port al realizar la lectura de uno de esos ports.

Ejemplo:

```
LD      A, 7FH        ; octeto de más peso del port 7FFEH
IN      A, (ØFEH)     ; lectura del port número 7FFEH
BIT     Ø, A          ; compruebe si la tecla SPACE está pulsada
JR      Z, BREAK      ; sí, ir a BREAK
```

Cuando las interrupciones están autorizadas, el subprograma de interrupción situado en la dirección 38H efectúa una lectura del teclado cada 2∅ milisegundos. Si una tecla está pulsada evalúa el código ASCII de esta tecla y la coloca en la posición de memoria situada en la dirección 2356∅. A continuación coloca a 1 el bit 5 del octeto situado en la dirección 23611 para indicar que se ha pulsado una tecla. Es este subprograma el que controla la temporización del teclado y la repetición automática de teclas.

Si se prohíben las interrupciones en un programa en lenguaje máquina y volvemos al monitor Basic, este último no aceptará ningún carácter pulsado en el teclado y no tendremos otra solución que desconectar el SPECTRUM. Deberemos pensar en esto al escribir subprogramas en lenguaje máquina que prohíban las interrupciones.

Cuando éstas están autorizadas podremos utilizar las informaciones suministradas por el subprograma de interrupción en un programa en ensamblador para obtener el código ASCII de la última tecla pulsada. Esto lo realiza el siguiente subprograma:

```
                00010  ; ************************************************************************
                00020  ;
                00030  ;  SUBPROGRAMA DE LECTURA DEL TECLADO
                00040  ;
                00050  ;  SALIDA: Z = VALE 1 SI NINGUNA TECLA HA SIDO PULSADA
                00060  ;          Z = VALE 0 SI UNA TECLA HA SIDO PULSADA
                00070  ;          Y A CONTIENE EL CODIGO ASCII DE LA TECLA
                00080  ;
                00090  ;  NINGUN REGISTRO ES MODIFICADO SALVO AF
                00100  ;
                00110  ; ************************************************************************
                00120  ;
0000 3A3B5C     00130 KEY  LD   A,(23611)   ; INDICADORES
0003 CB6F       00140      BIT  5,A         ; ¿TECLA PULSADA?
0005 C8         00150      RET  Z           ; NO RETORNO CON Z = 1
0006 CBAF       00160      RES  5,A         ; COLOCAR DE NUEVO EL INDICADOR A CERO
0008 323B5C     00170      LD   (23611),A   ;
000B 3A085C     00180      LD   A,(23560)   ; CODIGO ASCII DE LA TECLA PULSADA
000E C9         00190      RET              ; RETORNO CON Z = 0
0000            00200      END
```

Cuando deseemos conocer instantáneamente el estado del teclado deberemos leer las teclas pulsadas leyendo directamente los ports correspondientes. Es lo que empleamos en los juegos de acción rápida. Contrariamente, cuando queramos leer el teclado para conocer cierto número de caracteres ASCII, deberemos utilizar las informaciones suministradas por el subprograma de interrupción que controla la temporización del teclado. Recordemos que en este caso no deberemos modificar el registro IY utilizado en las interrupciones.

El subprograma siguiente emplea el subprograma KEY para leer una serie de caracteres ASCII como lo hace el interpretador Basic. La tecla < DELETE > (< CAPS SHIFT > + <∅>), sirve para borrar el último carácter pulsado y la tecla < ENTER > indica el fin de la línea. Este subprograma, además, utiliza los subprogramas AFI, TRAN, y TATR indicados en el párrafo 4.1.5.

```
                00010 ; ***************************************************************************
                00020 ;
                00030 ;   SUBPROGRAMA DE LECTURA DE UNA SERIE DE CARACTERES
                00040 ;   ASCII PULSADOS SOBRE EL TECLADO
                00050 ;
                00060 ;   ENTRADA: D  = NUMERO DE LINEA DONDE COMIENZA LA ENTRADA
                00070 ;            E  = NUMERO DE COLUMNA DONDE COMIENZA LA ENTRADA
                00080 ;            C  = NUMERO MAXIMO DE CARACTERES A LEER
                00090 ;            HL = DIRECCION DEL INICIO DE LA ZONA DE MEMORIA
                00100 ;                 DESTINADA A RECIBIR LA LINEA LEIDA
                00110 ;
                00120 ;   SALIDA: B = NUMERO DE CARACTERES LEIDOS
                00130 ;
                00140 ; ***************************************************************************
                00150 ;
0000 0600       00160 CUR     LD     B,0          ; NINGUN CARACTER LEIDO
0002 CD4A00     00170 ZY      CALL   KEY          ; LECTURA DE CARACTER
0005 281A       00180         JR     Z,ZW         ; SI NO HAY TECLA PULSADA
0007 FE0C       00190         CP     12           ; ¿ES ESTA LA TECLA DELETE?
0009 282A       00200         JR     Z,ZX         ; SI ES SI, TRATAR ESTE CASO
000B FE0D       00210         CP     13           ; ¿ES ESTO UN RETORNO DE CARRO?
000D C8         00220         RET    Z            ; SI ES SI, FIN DE LA ENTRADA
000E 77         00230         LD     (HL),A       ; GUARDAR CARACTER EN EL BUFFER
000F 78         00240         LD     A,B          ; NUMERO DE CARACTERES LEIDOS
0010 B9         00250         CP     C            ; ¿SUPERIOR A NUMERO MAXIMO?
0011 300E       00260         JR     NC,ZW        ; SI ES SI, NO ACEPTARLO
0013 7E         00270         LD     A,(HL)       ; CARACTER LEIDO
0014 23         00280         INC    HL           ; INCREMENTAR PUNTERO
0015 04         00290         INC    B            ; UN CARACTER LEIDO DE MAS
0016 CD5900     00300         CALL   AFI          ; VISUALIZAR ESTE CARACTER
0019 1C         00310         INC    E            ; COLUMNA SIGUIENTE
001A CB6B       00320         BIT    5,E          ; ¿ESTAMOS EN EL EXTREMO DE LA LINEA?
001C 2803       00330         JR     Z,ZW         ; SI NO
001E 14         00340         INC    D            ; LINEA SIGUIENTE
001F 1E00       00350         LD     E,0          ; COLOCARSE AL INICIO DE LA LINEA
0021 3AA500     00360 ZW      LD     A,(ATRB)     ; ATRIBUTO GRAFICO
0024 F5         00370         PUSH   AF           ; GUARDAR EL ATRIBUTO
0025 CBFF       00380         SET    7,A          ; PARPADEO
0027 32A500     00390         LD     (ATRB),A     ; NUEVO ATRIBUTO DE PARPADEO
002A 3E20       00400         LD     A,' '        ; ESPACIO
002C CD5900     00410         CALL   AFI          ; VISUALIZAR EL CURSOR
002F F1         00420         POP    AF           ; RECUPERAR ATRIBUTO
0030 32A500     00430         LD     (ATRB),A     ; RESTAURAR ATRIBUTO
0033 18CD       00440         JR     ZY           ; LEER EL CARACTER SIGUIENTE
0035 78         00450 ZX      LD     A,B          ; NUMERO DE CARACTERES LEIDOS
0036 B7         00460         OR     A            ; ¿NULO?
0037 28E8       00470         JR     Z,ZW         ; SI ES SI, NO TRATAR LA DELETE
0039 05         00480         DEC    B            ; UN CARACTER DE MENOS
003A 2B         00490         DEC    HL           ; DECREMENTAR PUNTERO
003B 3B20       00500         LD     A,' '        ; ESPACIO
003D CD5900     00510         CALL   AFI          ; BORRAR EL CURSOR
0040 1D         00520         DEC    E            ; COLUMNA PRECEDENTE
0041 CB7B       00530         BIT    7,E          ; ¿SALTO DE LINEA?
0043 28DC       00540         JR     Z,ZW         ; NO
0045 15         00550         DEC    D            ; LINEA PRECEDENTE
0046 1E1F       00560         LD     E,31         ; COLOCARSE AL FINAL DE LA LINEA
0048 18D7       00570         JR     ZW           ; CONTINUACION
```

4.5. Los mandos de juego

Las empuñaduras de juego de marca AGF, comercializadas por DIRECO, se colocan en paralelo con algunas teclas del teclado. El hecho de accionar uno de los elementos de la palanca equivale a la pulsación de la tecla correspondiente en el teclado. Así podremos leer el estado de los mandos de juego de la misma manera que lo hace-

mos para conocer el estado de las teclas correspondientes del teclado, es decir, mediante los ports de entrada/salida asociados, cuando se desee una lectura instantánea (en el caso de los juegos), o mediante el subprograma KEY. El cuadro siguiente resume esta correspondencia:

Elemento del mando de juego	*Izquierda jugador 1*	*Abajo jugador 1*	*Arriba jugador 1*	*Derecha jugador 1*	*Tira jugador 1*	*Izquierda jugador 2*	*Abajo jugador 2*	*Arriba jugador 2*	*Derecha jugador 2*	*Tira jugador 2*
Tecla del teclado	5	6	7	8	Ø	T	Y	U	I	P

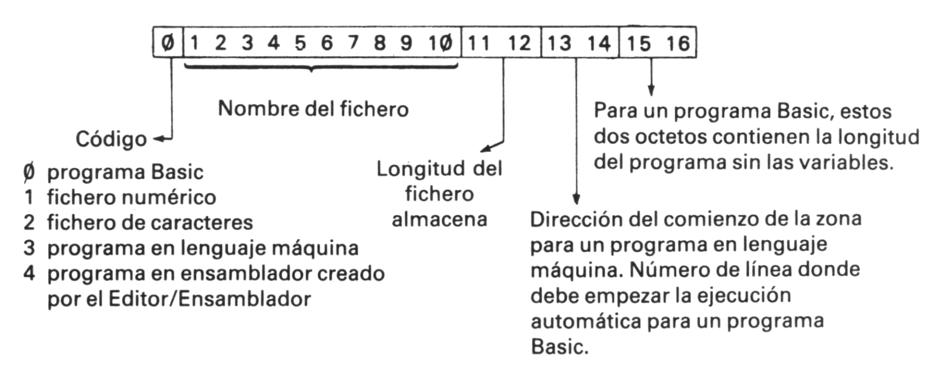
4.6. La interfase de los cassettes

El bit 3 del port 254, utilizado como salida (OUT), acciona la salida de cassette, lo que tiene como efecto enviar un pulso por la salida MIC del SPECTRUM.

Inversamente, el bit 6 utilizado como entrada (IN) recibe una señal presente en la entrada EAR.

Estos dos bits se emplean por los subprogramas de escritura y de lectura en cassette que están situados en ROM. Estos dos subprogramas permiten leer o escribir en el cassette un número de octetos fijados por el usuario a partir de una posición fijada por él.

El subprograma de escritura se sitúa en la dirección 4C2H. Como parámetros IX deberá contener la dirección de inicio de la zona a transferir y DE el número de octetos a transferir. El registro A tendrá el valor Ø, o el valor FFH, según se desee escribir el encabezamiento del fichero o el cuerpo del mismo. Los ficheros están efectivamente divididos en dos partes, registradas mediante un formato algo diferente. La primera parte, o encabezamiento del fichero, contiene diversas informaciones relativas al fichero. Ocupa 17 octetos organizados de la forma siguiente:



La segunda parte contiene los datos propiamente dichos del fichero. Para un fichero que memoriza un programa en lenguaje máquina, esta parte contendrá todos los octetos del programa.

La serie de instrucciones que permiten la escritura en cassette de una de estas partes será la siguiente:

```
LD      IX, dirección   ; dirección de inicio de la zona que contiene
                        ; los octetos a transferir.
LD      DE, número      ; número de octetos a transferir.
LD      A, código       ; ØØ o FFH, según la parte.
CALL    4C2H            ; escritura.
```

El subprograma de lectura situado en la dirección 556H necesita los mismos valores como parámetros. Además, será necesario que el indicador Carry se coloque a 1. Las siguientes instrucciones permiten la lectura de una de las partes del fichero:

```
LD      IX, dirección   ; dirección de la zona donde serán
                        ; transferidos los octetos leídos.
LD      DE, número      ; número de octetos a leer.
LD      A, código       ; ØØ o FFH, según la parte.
SCF                     ;
CALL    556H            ; lectura.
```

Anexo 1
Las bases de numeración

Para contar nos hemos habituado a emplear la base 1Ø o base decimal. Esta base utiliza 1Ø símbolos representados por las cifras del Ø al 9. Para contar una sucesión de objetos enumeramos sucesivamente cada una de las cifras del Ø al 9. Cuando deseamos aumentar el número 9 en una unidad, colocamos el símbolo 1 delante de la cifra 9 que es reemplazada por Ø, obtenemos así el número 1Ø. La primera es la cifra de las decenas. Representa diez veces su valor habitual. La segunda cifra es añadida al resultado para obtener el número final. Así, un número decimal de dos cifras que se escriba xy, vale: $1\emptyset * x + y$.

De la misma manera, un número decimal de tres cifras que se escriba xyz, vale $1\emptyset\emptyset * x + 1\emptyset * y + z$.

Generalizando, un número decimal de n cifras que se escriba $a_n a_{n-1} \dots a_1 a_\emptyset$ vale $a_n * 1\emptyset^n + a_{n-1} * 1\emptyset^{n-1} + \dots + a_1 * 1\emptyset + a^\emptyset$ donde $1\emptyset^k$, es igual al producto de k veces el número 1Ø.

$$1\emptyset^k = \underbrace{1\emptyset \times 1\emptyset \times \dots \times 1\emptyset \times 1\emptyset}_{\text{k veces}}$$

El número 1Ø de la base decimal representa el coeficiente multiplicativo que hay que utilizar para obtener el valor de una cifra, después del paso de una cifra a la posición adyacente a la izquierda en un número decimal. El número 1Ø es también igual al número de símbolos empleados en la base decimal.

La base 2, o base binaria, no utiliza más que dos símbolos representados por las cifras Ø y 1. El coeficiente multiplicativo evocado anteriormente vale 2.

En esta base los números de Ø a 1Ø en decimal se escriben respectivamente: Ø, 1, 1Ø, 11, 1ØØ, 1Ø1, 11Ø, 111, 1ØØØ, 1ØØ1, 1Ø1Ø.

Un número binario que se escriba $a_n\ a_{n-1} \dots a_1\ a_\emptyset$ vale en decimal:

$$a_n * 2^n + a_{n-1} * 2^{n-1} + \dots + a_1 * 2 + a_\emptyset.$$

Así: 1Ø1 vale en decimal: $1 * 2^2 + \emptyset * 2^1 + 1 = 4 + 1 = 5$

1Ø1Ø vale en decimal: $1 * 2^3 + \emptyset * 2^2 + 1 * 2 + \emptyset = 8 + 2 = 1\emptyset$

La base 2 que solamente posee dos símbolos se adapta bien a las tensiones con dos estados (Ø V o 5 V) de la informática. No obstante, es poco manejable porque los números importantes necesitan una cantidad de símbolos demasiado grande para ser representados en esta base. Debido a ello se prefiere la base 16 o hexadecimal. Esta base utiliza 16 símbolos representados por las cifras de Ø a 9 más las letras de A a F. El coeficiente multiplicativo es igual a 16.

En esta base los números de Ø a 2Ø en decimal se escriben respectivamente: Ø, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F, 1Ø, 11, 12, 13, 14.

Un número hexadecimal que se escriba $a_n a_{n-1} \ldots a_1 a_\emptyset$ vale en decimal:

$$a_n * 16^n + a_{n-1} * 16^{n-1} + \ldots + a_1 * 16 + a_\emptyset.$$

Así: 3C vale en decimal: $3 * 16 + 12 = 6\emptyset$

1E3A vale en decimal: $1 * 16^3 + 14 * 16^2 + 3 * 16 + 1\emptyset = 7738$

El interés de la base 16 proviene del hecho de que permite reagrupar cuatro símbolos binarios en un solo símbolo hexadecimal.

La siguiente tabla permite comprender mejor la correspondencia entre los números hexadecimales y los números binarios:

Decimal	*Binario*	*Hexadecimal*
Ø	ØØØØ	Ø
1	ØØØ1	1
2	ØØ1Ø	2
3	ØØ11	3
4	Ø1ØØ	4
5	Ø1Ø1	5
6	Ø11Ø	6
7	Ø111	7
8	1ØØØ	8
9	1ØØ1	9
1Ø	1Ø1Ø	A
11	1Ø11	B
12	11ØØ	C
13	11Ø1	D
14	111Ø	E
15	1111	F

Hay una correspondencia biunívoca entre un símbolo hexadecimal y un número binario cualquiera de 4 símbolos. De esto sacamos que la conversión de cualquier número binario en su representación

hexadecimal es muy rápida, agrupando los símbolos binarios de cuatro en cuatro.

Ejemplo:

1ØØ1	ØØ1Ø	111Ø	Ø111	binario
9	2	E	7	hexadecimal

La conversión de un número binario en su representación decimal es mucho menos rápida y necesita más cálculos.

Por lo tanto, en informática se utilizará sobre todo la base hexadecimal que permite conservar la estructura binaria de los números, siendo cuatro veces más concisos que con la base binaria.

Anexo 2
Lista de instrucciones del Z 8∅ clasificadas por códigos

NN designa un número de 16 bits; N un número de 8 bits; IND un índice, y DIS una distancia relativa.

Las instrucciones cuyo código empieza por CB, DD, ED, FD, estarán situadas al final del listado.

```
0000 00       00010     NOP
0001 01EEFF   00020     LD      BC,NN
0004 02       00030     LD      (BC),A
0005 03       00040     INC     BC
0006 04       00050     INC     B
0007 05       00060     DEC     B
0008 060A     00070     LD      B,N
000A 07       00080     RLCA
000B 08       00090     EX      AF,AF'
000C 09       00100     ADD     HL,BC
000D 0A       00110     LD      A,(BC)
000E 0B       00120     DEC     BC
000F 0C       00130     INC     C
0010 0D       00140     DEC     C
0011 0E0A     00150     LD      C,N
0013 0F       00160     RRCA
0014 102E     00170     DJNZ    $+DIS
0016 11EEFF   00180     LD      DE,NN
0019 12       00190     LD      (DE),A
001A 13       00200     INC     DE
001B 14       00210     INC     D
001C 15       00220     DEC     D
001D 160A     00230     LD      D,N
001F 17       00240     RLA
0020 182E     00250     JR      $+DIS
0022 19       00260     ADD     HL,DE
0023 1A       00270     LD      A,(DE)
0024 1B       00280     DEC     DE
0025 1C       00290     INC     E
0026 1D       00300     DEC     E
0027 1E0A     00310     LD      E,N
0029 1F       00320     RRA
002A 202E     00330     JR      NZ,$+DIS
002C 21EEFF   00340     LD      HL,NN
002F 22EEFF   00350     LD      (NN),HL
0032 23       00360     INC     HL
0033 24       00370     INC     H
0034 25       00380     DEC     H
0035 260A     00390     LD      H,N
0037 27       00400     DAA
0038 282E     00410     JR      Z,$+DIS
003A 29       00420     ADD     HL,HL
003B 2AEEFF   00430     LD      HL,(NN)
003E 2B       00440     DEC     HL
003F 2C       00450     INC     L
0040 2D       00460     DEC     L
```

```
0041 2E0A      00470          LD      L,N
0043 2F        00480          CPL
0044 302E      00490          JR      NC,$+DIS
0046 31EEFF    00500          LD      SP,NN
0049 32EEFF    00510          LD      (NN),A
004C 33        00520          INC     SP
004D 34        00530          INC     (HL)
004E 35        00540          DEC     (HL)
004F 360A      00550          LD      (HL),N
0051 37        00560          SCF
0052 382E      00570          JR      C,$+DIS
0054 39        00580          ADD     HL,SP
0055 3AEEFF    00590          LD      A,(NN)
0058 3B        00600          DEC     SP
0059 3C        00610          INC     A
005A 3D        00620          DEC     A
005B 3E0A      00630          LD      A,N
005D 3F        00640          CCF
005E 40        00650          LD      B,B
005F 41        00660          LD      B,C
0060 42        00670          LD      B,D
0061 43        00680          LD      B,E
0062 44        00690          LD      B,H
0063 45        00700          LD      B,L
0064 46        00710          LD      B,(HL)
0065 47        00720          LD      B,A
0066 48        00730          LD      C,B
0067 49        00740          LD      C,C
0068 4A        00750          LD      C,D
0069 4B        00760          LD      C,E
006A 4C        00770          LD      C,H
006B 4D        00780          LD      C,L
006C 4E        00790          LD      C,(HL)
006D 4F        00800          LD      C,A
006E 50        00810          LD      D,B
006F 51        00820          LD      D,C
0070 52        00830          LD      D,D
0071 53        00840          LD      D,E
0072 54        00850          LD      D,H
0073 55        00860          LD      D,L
0074 56        00870          LD      D,(HL)
0075 57        00880          LD      D,A
0076 58        00890          LD      E,B
0077 59        00900          LD      E,C
0078 5A        00910          LD      E,D
0079 5B        00920          LD      E,E
007A 5C        00930          LD      E,H
007B 5D        00940          LD      E,L
007C 5E        00950          LD      E,(HL)
007D 5F        00960          LD      E,A
007E 60        00970          LD      H,B
007F 61        00980          LD      H,C
0080 62        00990          LD      H,D
0081 63        01000          LD      H,E
0082 64        01010          LD      H,H
0083 65        01020          LD      H,L
0084 66        01030          LD      H,(HL)
0085 67        01040          LD      H,A
0086 68        01050          LD      L,B
0087 69        01060          LD      L,C
0088 6A        01070          LD      L,D
0089 6B        01080          LD      L,E
008A 6C        01090          LD      L,H
008B 6D        01100          LD      L,L
008C 6E        01110          LD      L,(HL)
008D 6F        01120          LD      L,A
008E 70        01130          LD      (HL),B
008F 71        01140          LD      (HL),C
```

```
0090 72        01150        LD      (HL),D
0091 73        01160        LD      (HL),E
0092 74        01170        LD      (HL),H
0093 75        01180        LD      (HL),L
0094 76        01190        HALT
0095 77        01200        LD      (HL),A
0096 78        01210        LD      A,B
0097 79        01220        LD      A,C
0098 7A        01230        LD      A,D
0099 7B        01240        LD      A,E
009A 7C        01250        LD      A,H
009B 7D        01260        LD      A,L
009C 7E        01270        LD      A,(HL)
009D 7F        01280        LD      A,A
009E 80        01290        ADD     A,B
009F 81        01300        ADD     A,C
00A0 82        01310        ADD     A,D
00A1 83        01320        ADD     A,E
00A2 84        01330        ADD     A,H
00A3 85        01340        ADD     A,L
00A4 86        01350        ADD     A,(HL)
00A5 87        01360        ADD     A,A
00A6 88        01370        ADC     A,B
00A7 89        01380        ADC     A,C
00A8 8A        01390        ADC     A,D
00A9 8B        01400        ADC     A,E
00AA 8C        01410        ADC     A,H
00AB 8D        01420        ADC     A,L
00AC 8E        01430        ADC     A,(HL)
00AD 8F        01440        ADC     A,A
00AE 90        01450        SUB     B
00AF 91        01460        SUB     C
00B0 92        01470        SUB     D
00B1 93        01480        SUB     E
00B2 94        01490        SUB     H
00B3 95        01500        SUB     L
00B4 96        01510        SUB     (HL)
00B5 97        01520        SUB     A
00B6 98        01530        SBC     A,B
00B7 99        01540        SBC     A,C
00B8 9A        01550        SBC     A,D
00B9 9B        01560        SBC     A,E
00BA 9C        01570        SBC     A,H
00BB 9D        01580        SBC     A,L
00BC 9E        01590        SBC     A,(HL)
00BD 9F        01600        SBC     A,A
00BE A0        01610        AND     B
00BF A1        01620        AND     C
00C0 A2        01630        AND     D
00C1 A3        01640        AND     E
00C2 A4        01650        AND     H
00C3 A5        01660        AND     L
00C4 A6        01670        AND     (HL)
00C5 A7        01680        AND     A
00C6 A8        01690        XOR     B
00C7 A9        01700        XOR     C
00C8 AA        01710        XOR     D
00C9 AB        01720        XOR     E
00CA AC        01730        XOR     H
00CB AD        01740        XOR     L
00CC AE        01750        XOR     (HL)
00CD AF        01760        XOR     A
00CE B0        01770        OR      B
00CF B1        01780        OR      C
00D0 B2        01790        OR      D
00D1 B3        01800        OR      E
00D2 B4        01810        OR      H
00D3 B5        01820        OR      L
```

```
00D4 B6        01830        OR      (HL)
00D5 B7        01840        OR      A
00D6 B8        01850        CP      B
00D7 B9        01860        CP      C
00D8 BA        01870        CP      D
00D9 BB        01880        CP      E
00DA BC        01890        CP      H
00DB BD        01900        CP      L
00DC BE        01910        CP      (HL)
00DD BF        01920        CP      A
00DE C0        01930        RET     NZ
00DF C1        01940        POP     BC
00E0 C2EEFF    01950        JP      NZ,NN
00E3 C3EEFF    01960        JP      NN
00E6 C4EEFF    01970        CALL    NZ,NN
00E9 C5        01980        PUSH    BC
00EA C60A      01990        ADD     A,N
00EC C7        02000        RST     0
00ED C8        02010        RET     Z
00EE C9        02020        RET
00EF CAEEFF    02030        JP      Z,NN
00F2 CCEEFF    02040        CALL    Z,NN
00F5 CDEEFF    02050        CALL    NN
00F8 CE0A      02060        ADC     A,N
00FA CF        02070        RST     8
00FB D0        02080        RET     NC
00FC D1        02090        POP     DE
00FD D2EEFF    02100        JP      NC,NN
0100 D30A      02110        OUT     (N),A
0102 D4EEFF    02120        CALL    NC,NN
0105 D5        02130        PUSH    DE
0106 D60A      02140        SUB     N
0108 D7        02150        RST     10H
0109 D8        02160        RET     C
010A D9        02170        EXX
010B DAEEFF    02180        JP      C,NN
010E DB0A      02190        IN      A,(N)
0110 DCEEFF    02200        CALL    C,NN
0113 DE0A      02210        SBC     A,N
0115 DF        02220        RST     18H
0116 E0        02230        RET     PO
0117 E1        02240        POP     HL
0118 E2EEFF    02250        JP      PO,NN
011B E3        02260        EX      (SP),HL
011C E4EEFF    02270        CALL    PO,NN
011F E5        02280        PUSH    HL
0120 E60A      02290        AND     N
0122 E7        02300        RST     20H
0123 E8        02310        RET     PE
0124 E9        02320        JP      (HL)
0125 EAEEFF    02330        JP      PE,NN
0128 EB        02340        EX      DE,HL
0129 ECEEFF    02350        CALL    PE,NN
012C EE0A      02360        XOR     N
012E EF        02370        RST     28H
012F F0        02380        RET     P
0130 F1        02390        POP     AF
0131 F2EEFF    02400        JP      P,NN
0134 F3        02410        DI
0135 F4EEFF    02420        CALL    P,NN
0138 F5        02430        PUSH    AF
0139 F60A      02440        OR      N
013B F7        02450        RST     30H
013C F8        02460        RET     M
013D F9        02470        LD      SP,HL
013E FAEEFF    02480        JP      M,NN
0141 FB        02490        EI
0142 FCEEFF    02500        CALL    M,NN
```

```
0145 FE0A       02510          CP       N
0147 FF         02520          RST      38H
0148 CB00       02530          RLC      B
014A CB01       02540          RLC      C
014C CB02       02550          RLC      D
014E CB03       02560          RLC      E
0150 CB04       02570          RLC      H
0152 CB05       02580          RLC      L
0154 CB06       02590          RLC      (HL)
0156 CB07       02600          RLC      A
0158 CB08       02610          RRC      B
015A CB09       02620          RRC      C
015C CB0A       02630          RRC      D
015E CB0B       02640          RRC      E
0160 CB0C       02650          RRC      H
0162 CB0D       02660          RRC      L
0164 CB0E       02670          RRC      (HL)
0166 CB0F       02680          RRC      A
0168 CB10       02690          RL       B
016A CB11       02700          RL       C
016C CB12       02710          RL       D
016E CB13       02720          RL       E
0170 CB14       02730          RL       H
0172 CB15       02740          RL       L
0174 CB16       02750          RL       (HL)
0176 CB17       02760          RL       A
0178 CB18       02770          RR       B
017A CB19       02780          RR       C
017C CB1A       02790          RR       D
017E CB1B       02800          RR       E
0180 CB1C       02810          RR       H
0182 CB1D       02820          RR       L
0184 CB1E       02830          RR       (HL)
0186 CB1F       02840          RR       A
0188 CB20       02850          SLA      B
018A CB21       02860          SLA      C
018C CB22       02870          SLA      D
018E CB23       02880          SLA      E
0190 CB24       02890          SLA      H
0192 CB25       02900          SLA      L
0194 CB26       02910          SLA      (HL)
0196 CB27       02920          SLA      A
0198 CB28       02930          SRA      B
019A CB29       02940          SRA      C
019C CB2A       02950          SRA      D
019E CB2B       02960          SRA      E
01A0 CB2C       02970          SRA      H
01A2 CB2D       02980          SRA      L
01A4 CB2E       02990          SRA      (HL)
01A6 CB2F       03000          SRA      A
01A8 CB38       03010          SRL      B
01AA CB39       03020          SRL      C
01AC CB3A       03030          SRL      D
01AE CB3B       03040          SRL      E
01B0 CB3C       03050          SRL      H
01B2 CB3D       03060          SRL      L
01B4 CB3E       03070          SRL      (HL)
01B6 CB3F       03080          SRL      A
01B8 CB40       03090          BIT      0,B
01BA CB41       03100          BIT      0,C
01BC CB42       03110          BIT      0,D
01BE CB43       03120          BIT      0,E
01C0 CB44       03130          BIT      0,H
01C2 CB45       03140          BIT      0,L
01C4 CB46       03150          BIT      0,(HL)
01C6 CB47       03160          BIT      0,A
01C8 CB48       03170          BIT      1,B
01CA CB49       03180          BIT      1,C
```

```
01CC CB4A        03190            BIT     1,D
01CE CB4B        03200            BIT     1,E
01D0 CB4C        03210            BIT     1,H
01D2 CB4D        03220            BIT     1,L
01D4 CB4E        03230            BIT     1,(HL)
01D6 CB4F        03240            BIT     1,A
01D8 CB50        03250            BIT     2,B
01DA CB51        03260            BIT     2,C
01DC CB52        03270            BIT     2,D
01DE CB53        03280            BIT     2,E
01E0 CB54        03290            BIT     2,H
01E2 CB55        03300            BIT     2,L
01E4 CB56        03310            BIT     2,(HL)
01E6 CB57        03320            BIT     2,A
01E8 CB58        03330            BIT     3,B
01EA CB59        03340            BIT     3,C
01EC CB5A        03350            BIT     3,D
01EE CB5B        03360            BIT     3,E
01F0 CB5C        03370            BIT     3,H
01F2 CB5D        03380            BIT     3,L
01F4 CB5E        03390            BIT     3,(HL)
01F6 CB5F        03400            BIT     3,A
01F8 CB60        03410            BIT     4,B
01FA CB61        03420            BIT     4,C
01FC CB62        03430            BIT     4,D
01FE CB63        03440            BIT     4,E
0200 CB64        03450            BIT     4,H
0202 CB65        03460            BIT     4,L
0204 CB66        03470            BIT     4,(HL)
0206 CB67        03480            BIT     4,A
0208 CB68        03490            BIT     5,B
020A CB69        03500            BIT     5,C
020C CB6A        03510            BIT     5,D
020E CB6B        03520            BIT     5,E
0210 CB6C        03530            BIT     5,H
0212 CB6D        03540            BIT     5,L
0214 CB6E        03550            BIT     5,(HL)
0216 CB6F        03560            BIT     5,A
0218 CB70        03570            BIT     6,B
021A CB71        03580            BIT     6,C
021C CB72        03590            BIT     6,D
021E CB73        03600            BIT     6,E
0220 CB74        03610            BIT     6,H
0222 CB75        03620            BIT     6,L
0224 CB76        03630            BIT     6,(HL)
0226 CB77        03640            BIT     6,A
0228 CB78        03650            BIT     7,B
022A CB79        03660            BIT     7,C
022C CB7A        03670            BIT     7,D
022E CB7B        03680            BIT     7,E
0230 CB7C        03690            BIT     7,H
0232 CB7D        03700            BIT     7,L
0234 CB7E        03710            BIT     7,(HL)
0236 CB7F        03720            BIT     7,A
0238 CB80        03730            RES     0,B
023A CB81        03740            RES     0,C
023C CB82        03750            RES     0,D
023E CB83        03760            RES     0,E
0240 CB84        03770            RES     0,H
0242 CB85        03780            RES     0,L
0244 CB86        03790            RES     0,(HL)
0246 CB87        03800            RES     0,A
0248 CB88        03810            RES     1,B
024A CB89        03820            RES     1,C
024C CB8A        03830            RES     1,D
024E CB8B        03840            RES     1,E
0250 CB8C        03850            RES     1,H
0252 CB8D        03860            RES     1,L
```

```
0254 CB8E       03870          RES      1,(HL)
0256 CB8F       03880          RES      1,A
0258 CB90       03890          RES      2,B
025A CB91       03900          RES      2,C
025C CB92       03910          RES      2,D
025E CB93       03920          RES      2,E
0260 CB94       03930          RES      2,H
0262 CB95       03940          RES      2,L
0264 CB96       03950          RES      2,(HL)
0266 CB97       03960          RES      2,A
0268 CB98       03970          RES      3,B
026A CB99       03980          RES      3,C
026C CB9A       03990          RES      3,D
026E CB9B       04000          RES      3,E
0270 CB9C       04010          RES      3,H
0272 CB9D       04020          RES      3,L
0274 CB9E       04030          RES      3,(HL)
0276 CB9F       04040          RES      3,A
0278 CBA0       04050          RES      4,B
027A CBA1       04060          RES      4,C
027C CBA2       04070          RES      4,D
027E CBA3       04080          RES      4,E
0280 CBA4       04090          RES      4,H
0282 CBA5       04100          RES      4,L
0284 CBA6       04110          RES      4,(HL)
0286 CBA7       04120          RES      4,A
0288 CBA8       04130          RES      5,B
028A CBA9       04140          RES      5,C
028C CBAA       04150          RES      5,D
028E CBAB       04160          RES      5,E
0290 CBAC       04170          RES      5,H
0292 CBAD       04180          RES      5,L
0294 CBAE       04190          RES      5,(HL)
0296 CBAF       04200          RES      5,A
0298 CBB0       04210          RES      6,B
029A CBB1       04220          RES      6,C
029C CBB2       04230          RES      6,D
029E CBB3       04240          RES      6,E
02A0 CBB4       04250          RES      6,H
02A2 CBB5       04260          RES      6,L
02A4 CBB6       04270          RES      6,(HL)
02A6 CBB7       04280          RES      6,A
02A8 CBB8       04290          RES      7,B
02AA CBB9       04300          RES      7,C
02AC CBBA       04310          RES      7,D
02AE CBBB       04320          RES      7,E
02B0 CBBC       04330          RES      7,H
02B2 CBBD       04340          RES      7,L
02B4 CBBE       04350          RES      7,(HL)
02B6 CBBF       04360          RES      7,A
02B8 CBC0       04370          SET      0,B
02BA CBC1       04380          SET      0,C
02BC CBC2       04390          SET      0,D
02BE CBC3       04400          SET      0,E
02C0 CBC4       04410          SET      0,H
02C2 CBC5       04420          SET      0,L
02C4 CBC6       04430          SET      0,(HL)
02C6 CBC7       04440          SET      0,A
02C8 CBC8       04450          SET      1,B
02CA CBC9       04460          SET      1,C
02CC CBCA       04470          SET      1,D
02CE CBCB       04480          SET      1,E
02D0 CBCC       04490          SET      1,H
02D2 CBCD       04500          SET      1,L
02D4 CBCE       04510          SET      1,(HL)
02D6 CBCF       04520          SET      1,A
02D8 CBD0       04530          SET      2,B
02DA CBD1       04540          SET      2,C
```

```
02DC CBD2      04550        SET      2,D
02DE CBD3      04560        SET      2,E
02E0 CBD4      04570        SET      2,H
02E2 CBD5      04580        SET      2,L
02E4 CBD6      04590        SET      2,(HL)
02E6 CBD7      04600        SET      2,A
02E8 CBD8      04610        SET      3,B
02EA CBD9      04620        SET      3,C
02EC CBDA      04630        SET      3,D
02EE CBDB      04640        SET      3,E
02F0 CBDC      04650        SET      3,H
02F2 CBDD      04660        SET      3,L
02F4 CBDE      04670        SET      3,(HL)
02F6 CBDF      04680        SET      3,A
02F8 CBE0      04690        SET      4,B
02FA CBE1      04700        SET      4,C
02FC CBE2      04710        SET      4,D
02FE CBE3      04720        SET      4,E
0300 CBE4      04730        SET      4,H
0302 CBE5      04740        SET      4,L
0304 CBE6      04750        SET      4,(HL)
0306 CBE7      04760        SET      4,A
0308 CBE8      04770        SET      5,B
030A CBE9      04780        SET      5,C
030C CBEA      04790        SET      5,D
030E CBEB      04800        SET      5,E
0310 CBEC      04810        SET      5,H
0312 CBED      04820        SET      5,L
0314 CBEE      04830        SET      5,(HL)
0316 CBEF      04840        SET      5,A
0318 CBF0      04850        SET      6,B
031A CBF1      04860        SET      6,C
031C CBF2      04870        SET      6,D
031E CBF3      04880        SET      6,E
0320 CBF4      04890        SET      6,H
0322 CBF5      04900        SET      6,L
0324 CBF6      04910        SET      6,(HL)
0326 CBF7      04920        SET      6,A
0328 CBF8      04930        SET      7,B
032A CBF9      04940        SET      7,C
032C CBFA      04950        SET      7,D
032E CBFB      04960        SET      7,E
0330 CBFC      04970        SET      7,H
0332 CBFD      04980        SET      7,L
0334 CBFE      04990        SET      7,(HL)
0336 CBFF      05000        SET      7,A
0338 DD09      05010        ADD      IX,BC
033A DD19      05020        ADD      IX,DE
033C DD21EEFF  05030        LD       IX,NN
0340 DD22EEFF  05040        LD       (NN),IX
0344 DD23      05050        INC      IX
0346 DD29      05060        ADD      IX,IX
0348 DD2AEEFF  05070        LD       IX,(NN)
034C DD2B      05080        DEC      IX
034E DD3405    05090        INC      (IX+IND)
0351 DD3505    05100        DEC      (IX+IND)
0354 DD36050A  05110        LD       (IX+IND),N
0358 DD39      05120        ADD      IX,SP
035A DD4605    05130        LD       B,(IX+IND)
035D DD4E05    05140        LD       C,(IX+IND)
0360 DD5605    05150        LD       D,(IX+IND)
0363 DD5E05    05160        LD       E,(IX+IND)
0366 DD6605    05170        LD       H,(IX+IND)
0369 DD6E05    05180        LD       L,(IX+IND)
036C DD7005    05190        LD       (IX+IND),B
036F DD7105    05200        LD       (IX+IND),C
0372 DD7205    05210        LD       (IX+IND),D
0375 DD7305    05220        LD       (IX+IND),E
```

```
0378 DD7405     05230          LD      (IX+IND),H
037B DD7505     05240          LD      (IX+IND),L
037E DD7705     05250          LD      (IX+IND),A
0381 DD7E05     05260          LD      A,(IX+IND)
0384 DD8605     05270          ADD     A,(IX+IND)
0387 DD8E05     05280          ADC     A,(IX+IND)
038A DD9605     05290          SUB     (IX+IND)
038D DD9E05     05300          SBC     A,(IX+IND)
0390 DDA605     05310          AND     (IX+IND)
0393 DDAE05     05320          XOR     (IX+IND)
0396 DDB605     05330          OR      (IX+IND)
0399 DDBE05     05340          CP      (IX+IND)
039C DDE1       05350          POP     IX
039E DDE3       05360          EX      (SP),IX
03A0 DDE5       05370          PUSH    IX
03A2 DDE9       05380          JP      (IX)
03A4 DDF9       05390          LD      SP,IX
03A6 DDCB0506 05400            RLC     (IX+IND)
03AA DDCB050E 05410            RRC     (IX+IND)
03AE DDCB0516 05420            RL      (IX+IND)
03B2 DDCB051E 05430            RR      (IX+IND)
03B6 DDCB0526 05440            SLA     (IX+IND)
03BA DDCB052E 05450            SRA     (IX+IND)
03BE DDCB053E 05460            SRL     (IX+IND)
03C2 DDCB0546 05470            BIT     0,(IX+IND)
03C6 DDCB054E 05480            BIT     1,(IX+IND)
03CA DDCB0556 05490            BIT     2,(IX+IND)
03CE DDCB055E 05500            BIT     3,(IX+IND)
03D2 DDCB0566 05510            BIT     4,(IX+IND)
03D6 DDCB056E 05520            BIT     5,(IX+IND)
03DA DDCB0576 05530            BIT     6,(IX+IND)
03DE DDCB057E 05540            BIT     7,(IX+IND)
03E2 DDCB0586 05550            RES     0,(IX+IND)
03E6 DDCB058E 05560            RES     1,(IX+IND)
03EA DDCB0596 05570            RES     2,(IX+IND)
03EE DDCB059E 05580            RES     3,(IX+IND)
03F2 DDCB05A6 05590            RES     4,(IX+IND)
03F6 DDCB05AE 05600            RES     5,(IX+IND)
03FA DDCB05B6 05610            RES     6,(IX+IND)
03FE DDCB05BE 05620            RES     7,(IX+IND)
0402 DDCB05C6 05630            SET     0,(IX+IND)
0406 DDCB05CE 05640            SET     1,(IX+IND)
040A DDCB05D6 05650            SET     2,(IX+IND)
040E DDCB05DE 05660            SET     3,(IX+IND)
0412 DDCB05E6 05670            SET     4,(IX+IND)
0416 DDCB05EE 05680            SET     5,(IX+IND)
041A DDCB05F6 05690            SET     6,(IX+IND)
041E DDCB05FE 05700            SET     7,(IX+IND)
0422 ED40       05710          IN      B,(C)
0424 ED41       05720          OUT     (C),B
0426 ED42       05730          SBC     HL,BC
0428 ED43EEFF 05740            LD      (NN),BC
042C ED44       05750          NEG
042E ED45       05760          RETN
0430 ED46       05770          IM      0
0432 ED47       05780          LD      I,A
0434 ED48       05790          IN      C,(C)
0436 ED49       05800          OUT     (C),C
0438 ED4A       05810          ADC     HL,BC
043A ED4BEEFF 05820            LD      BC,(NN)
043E ED4D       05830          RETI
0440 ED4F       05840          LD      R,A
0442 ED50       05850          IN      D,(C)
0444 ED51       05860          OUT     (C),D
0446 ED52       05870          SBC     HL,DE
0448 ED53EEFF 05880            LD      (NN),DE
044C ED56       05890          IM      1
044E ED57       05900          LD      A,I
```

```
0450 ED58       05910          IN      E,(C)
0452 ED59       05920          OUT     (C),E
0454 ED5A       05930          ADC     HL,DE
0456 ED5BEEFF   05940          LD      DE,(NN)
045A ED5E       05950          IM      2
045C ED5F       05960          LD      A,R
045E ED60       05970          IN      H,(C)
0460 ED61       05980          OUT     (C),H
0462 ED62       05990          SBC     HL,HL
0464 ED67       06000          RRD
0466 ED68       06010          IN      L,(C)
0468 ED69       06020          OUT     (C),L
046A ED6A       06030          ADC     HL,HL
046C ED6F       06040          RLD
046E ED72       06050          SBC     HL,SP
0470 ED73EEFF   06060          LD      (NN),SP
0474 ED78       06070          IN      A,(C)
0476 ED79       06080          OUT     (C),A
0478 ED7A       06090          ADC     HL,SP
047A ED7BEEFF   06100          LD      SP,(NN)
047E EDA0       06110          LDI
0480 EDA1       06120          CPI
0482 EDA2       06130          INI
0484 EDA3       06140          OUTI
0486 EDA8       06150          LDD
0488 EDA9       06160          CPD
048A EDAA       06170          IND
048C EDAB       06180          OUTD
048E EDB0       06190          LDIR
0490 EDB1       06200          CPIR
0492 EDB2       06210          INIR
0494 EDB3       06220          OTIR
0496 EDB8       06230          LDDR
0498 EDB9       06240          CPDR
049A EDBA       06250          INDR
049C EDBB       06260          OTDR
049E FD09       06270          ADD     IY,BC
04A0 FD19       06280          ADD     IY,DE
04A2 FD21EEFF   06290          LD      IY,NN
04A6 FD22EEFF   06300          LD      (NN),IY
04AA FD23       06310          INC     IY
04AC FD29       06320          ADD     IY,IY
04AE FD2AEEFF   06330          LD      IY,(NN)
04B2 FD2B       06340          DEC     IY
04B4 FD3405     06350          INC     (IY+IND)
04B7 FD3505     06360          DEC     (IY+IND)
04BA FD36050A   06370          LD      (IY+IND),N
04BE FD39       06380          ADD     IY,SP
04C0 FD4605     06390          LD      B,(IY+IND)
04C3 FD4E05     06400          LD      C,(IY+IND)
04C6 FD5605     06410          LD      D,(IY+IND)
04C9 FD5E05     06420          LD      E,(IY+IND)
04CC FD6605     06430          LD      H,(IY+IND)
04CF FD6E05     06440          LD      L,(IY+IND)
04D2 FD7005     06450          LD      (IY+IND),B
04D5 FD7105     06460          LD      (IY+IND),C
04D8 FD7205     06470          LD      (IY+IND),D
04DB FD7305     06480          LD      (IY+IND),E
04DE FD7405     06490          LD      (IY+IND),H
04E1 FD7505     06500          LD      (IY+IND),L
04E4 FD7705     06510          LD      (IY+IND),A
04E7 FD7E05     06520          LD      A,(IY+IND)
04EA FD8605     06530          ADD     A,(IY+IND)
04ED FD8E05     06540          ADC     A,(IY+IND)
04F0 FD9605     06550          SUB     (IY+IND)
04F3 FD9E05     06560          SBC     A,(IY+IND)
04F6 FDA605     06570          AND     (IY+IND)
04F9 FDAE05     06580          XOR     (IY+IND)
```

```
04FC FDB605    06590          OR      (IY+IND)
04FF FDBE05    06600          CP      (IY+IND)
0502 FDE1      06610          POP     IY
0504 FDE3      06620          EX      (SP),IY
0506 FDE5      06630          PUSH    IY
0508 FDE9      06640          JP      (IY)
050A FDF9      06650          LD      SP,IY
050C FDCB0506  06660          RLC     (IY+IND)
0510 FDCB050E  06670          RRC     (IY+IND)
0514 FDCB0516  06680          RL      (IY+IND)
0518 FDCB051E  06690          RR      (IY+IND)
051C FDCB0526  06700          SLA     (IY+IND)
0520 FDCB052E  06710          SRA     (IY+IND)
0524 FDCB053E  06720          SRL     (IY+IND)
0528 FDCB0546  06730          BIT     0,(IY+IND)
052C FDCB054E  06740          BIT     1,(IY+IND)
0530 FDCB0556  06750          BIT     2,(IY+IND)
0534 FDCB055E  06760          BIT     3,(IY+IND)
0538 FDCB0566  06770          BIT     4,(IY+IND)
053C FDCB056E  06780          BIT     5,(IY+IND)
0540 FDCB0576  06790          BIT     6,(IY+IND)
0544 FDCB057E  06800          BIT     7,(IY+IND)
0548 FDCB0586  06810          RES     0,(IY+IND)
054C FDCB058E  06820          RES     1,(IY+IND)
0550 FDCB0596  06830          RES     2,(IY+IND)
0554 FDCB059E  06840          RES     3,(IY+IND)
0558 FDCB05A6  06850          RES     4,(IY+IND)
055C FDCB05AE  06860          RES     5,(IY+IND)
0560 FDCB05B6  06870          RES     6,(IY+IND)
0564 FDCB05BE  06880          RES     7,(IY+IND)
0568 FDCB05C6  06890          SET     0,(IY+IND)
056C FDCB05CE  06900          SET     1,(IY+IND)
0570 FDCB05D6  06910          SET     2,(IY+IND)
0574 FDCB05DE  06920          SET     3,(IY+IND)
0578 FDCB05E6  06930          SET     4,(IY+IND)
057C FDCB05EE  06940          SET     5,(IY+IND)
0580 FDCB05F6  06950          SET     6,(IY+IND)
0584 FDCB05FE  06960          SET     7,(IY+IND)
FFEE           06970 NN       EQU     0FFEEH
0005           06980 IND      EQU     5
000A           06990 N        EQU     10
0030           07000 DIS      EQU     30H
0000           07010          END
```

Anexo 3
Lista de instrucciones del Z 8Ø clasificadas por mnemotécnicos

NN designa un número de 16 bits; N un número de 8 bits; IND un índice y DIS una distancia relativa.

Dirección	*Código*	*Línea*	*Mnemotécnico*	
0000	8E	00010	ADC	A,(HL)
0001	DD8E05	00020	ADC	A,(IX+IND)
0004	FD8E05	00030	ADC	A,(IY+IND)
0007	8F	00040	ADC	A,A
0008	88	00050	ADC	A,B
0009	89	00060	ADC	A,C
000A	8A	00070	ADC	A,D
000B	8B	00080	ADC	A,E
000C	8C	00090	ADC	A,H
000D	8D	00100	ADC	A,L
000E	CE10	00110	ADC	A,N
0010	ED4A	00120	ADC	HL,BC
0012	ED5A	00130	ADC	HL,DE
0014	ED6A	00140	ADC	HL,HL
0016	ED7A	00150	ADC	HL,SP
0018	86	00160	ADD	A,(HL)
0019	DD8605	00170	ADD	A,(IX+IND)
001C	FD8605	00180	ADD	A,(IY+IND)
001F	87	00190	ADD	A,A
0020	80	00200	ADD	A,B
0021	81	00210	ADD	A,C
0022	82	00220	ADD	A,D
0023	83	00230	ADD	A,E
0024	84	00240	ADD	A,H
0025	85	00250	ADD	A,L
0026	C610	00260	ADD	A,N
0028	09	00270	ADD	HL,BC
0029	19	00280	ADD	HL,DE
002A	29	00290	ADD	HL,HL
002B	39	00300	ADD	HL,SP
002C	DD09	00310	ADD	IX,BC
002E	DD19	00320	ADD	IX,DE
0030	DD29	00330	ADD	IX,IX
0032	DD39	00340	ADD	IX,SP
0034	FD09	00350	ADD	IY,BC
0036	FD19	00360	ADD	IY,DE
0038	FD29	00370	ADD	IY,IY
003A	FD39	00380	ADD	IY,SP
003C	A6	00390	AND	(HL)
003D	DDA605	00400	AND	(IX+IND)
0040	FDA605	00410	AND	(IY+IND)
0043	A7	00420	AND	A
0044	A0	00430	AND	B
0045	A1	00440	AND	C
0046	A2	00450	AND	D

```
0047 A3        00460          AND     E
0048 A4        00470          AND     H
0049 A5        00480          AND     L
004A E610      00490          AND     N
004C CB46      00500          BIT     0,(HL)
004E DDCB0546  00510          BIT     0,(IX+IND)
0052 FDCB0546  00520          BIT     0,(IY+IND)
0056 CB47      00530          BIT     0,A
0058 CB40      00540          BIT     0,B
005A CB41      00550          BIT     0,C
005C CB42      00560          BIT     0,D
005E CB43      00570          BIT     0,E
0060 CB44      00580          BIT     0,H
0062 CB45      00590          BIT     0,L
0064 CB4E      00600          BIT     1,(HL)
0066 DDCB054E  00610          BIT     1,(IX+IND)
006A FDCB054E  00620          BIT     1,(IY+IND)
006E CB4F      00630          BIT     1,A
0070 CB48      00640          BIT     1,B
0072 CB49      00650          BIT     1,C
0074 CB4A      00660          BIT     1,D
0076 CB4B      00670          BIT     1,E
0078 CB4C      00680          BIT     1,H
007A CB4D      00690          BIT     1,L
007C CB56      00700          BIT     2,(HL)
007E DDCB0556  00710          BIT     2,(IX+IND)
0082 FDCB0556  00720          BIT     2,(IY+IND)
0086 CB57      00730          BIT     2,A
0088 CB50      00740          BIT     2,B
008A CB51      00750          BIT     2,C
008C CB52      00760          BIT     2,D
008E CB53      00770          BIT     2,E
0090 CB54      00780          BIT     2,H
0092 CB55      00790          BIT     2,L
0094 CB5E      00800          BIT     3,(HL)
0096 DDCB055E  00810          BIT     3,(IX+IND)
009A FDCB055E  00820          BIT     3,(IY+IND)
009E CB5F      00830          BIT     3,A
00A0 CB58      00840          BIT     3,B
00A2 CB59      00850          BIT     3,C
00A4 CB5A      00860          BIT     3,D
00A6 CB5B      00870          BIT     3,E
00A8 CB5C      00880          BIT     3,H
00AA CB5D      00890          BIT     3,L
00AC CB66      00900          BIT     4,(HL)
00AE DDCB0566  00910          BIT     4,(IX+IND)
00B2 FDCB0566  00920          BIT     4,(IY+IND)
00B6 CB67      00930          BIT     4,A
00B8 CB60      00940          BIT     4,B
00BA CB61      00950          BIT     4,C
00BC CB62      00960          BIT     4,D
00BE CB63      00970          BIT     4,E
00C0 CB64      00980          BIT     4,H
00C2 CB65      00990          BIT     4,L
00C4 CB6E      01000          BIT     5,(HL)
00C6 DDCB056E  01010          BIT     5,(IX+IND)
00CA FDCB056E  01020          BIT     5,(IY+IND)
00CE CB6F      01030          BIT     5,A
00D0 CB68      01040          BIT     5,B
00D2 CB69      01050          BIT     5,C
00D4 CB6A      01060          BIT     5,D
00D6 CB6B      01070          BIT     5,E
00D8 CB6C      01080          BIT     5,H
00DA CB6D      01090          BIT     5,L
00DC CB76      01100          BIT     6,(HL)
00DE DDCB0576  01110          BIT     6,(IX+IND)
00E2 FDCB0576  01120          BIT     6,(IY+IND)
00E6 CB77      01130          BIT     6,A
```

```
00E8 CB70      01140        BIT     6,B
00EA CB71      01150        BIT     6,C
00EC CB72      01160        BIT     6,D
00EE CB73      01170        BIT     6,E
00F0 CB74      01180        BIT     6,H
00F2 CB75      01190        BIT     6,L
00F4 CB7E      01200        BIT     7,(HL)
00F6 DDCB057E  01210        BIT     7,(IX+IND)
00FA FDCB057E  01220        BIT     7,(IY+IND)
00FE CB7F      01230        BIT     7,A
0100 CB78      01240        BIT     7,B
0102 CB79      01250        BIT     7,C
0104 CB7A      01260        BIT     7,D
0106 CB7B      01270        BIT     7,E
0108 CB7C      01280        BIT     7,H
010A CB7D      01290        BIT     7,L
010C DCEEFF    01300        CALL    C,NN
010F FCEEFF    01310        CALL    M,NN
0112 D4EEFF    01320        CALL    NC,NN
0115 CDEEFF    01330        CALL    NN
0118 C4EEFF    01340        CALL    NZ,NN
011B F4EEFF    01350        CALL    P,NN
011E ECEEFF    01360        CALL    PE,NN
0121 E4EEFF    01370        CALL    PO,NN
0124 CCEEFF    01380        CALL    Z,NN
0127 3F        01390        CCF
0128 BE        01400        CP      (HL)
0129 DDBE05    01410        CP      (IX+IND)
012C FDBE05    01420        CP      (IY+IND)
012F BF        01430        CP      A
0130 B8        01440        CP      B
0131 B9        01450        CP      C
0132 BA        01460        CP      D
0133 BB        01470        CP      E
0134 BC        01480        CP      H
0135 BD        01490        CP      L
0136 FE10      01500        CP      N
0138 EDA9      01510        CPD
013A EDB9      01520        CPDR
013C EDA1      01530        CPI
013E EDB1      01540        CPIR
0140 2F        01550        CPL
0141 27        01560        DAA
0142 35        01570        DEC     (HL)
0143 DD3505    01580        DEC     (IX+IND)
0146 FD3505    01590        DEC     (IY+IND)
0149 3D        01600        DEC     A
014A 05        01610        DEC     B
014B 0B        01620        DEC     BC
014C 0D        01630        DEC     C
014D 15        01640        DEC     D
014E 1B        01650        DEC     DE
014F 1D        01660        DEC     E
0150 25        01670        DEC     H
0151 2B        01680        DEC     HL
0152 DD2B      01690        DEC     IX
0154 FD2B      01700        DEC     IY
0156 2D        01710        DEC     L
0157 3B        01720        DEC     SP
0158 F3        01730        DI
0159 102E      01740        DJNZ    $+DIS
015B FB        01750        EI
015C E3        01760        EX      (SP),HL
015D DDE3      01770        EX      (SP),IX
015F FDE3      01780        EX      (SP),IY
0161 08        01790        EX      AF,AF'
0162 EB        01800        EX      DE,HL
0163 D9        01810        EXX
```

```
0164 76         01820        HALT
0165 ED46       01830        IM      0
0167 ED56       01840        IM      1
0169 ED5E       01850        IM      2
016B ED78       01860        IN      A,(C)
016D DB10       01870        IN      A,(N)
016F ED40       01880        IN      B,(C)
0171 ED48       01890        IN      C,(C)
0173 ED50       01900        IN      D,(C)
0175 ED58       01910        IN      E,(C)
0177 ED60       01920        IN      H,(C)
0179 ED68       01930        IN      L,(C)
017B 34         01940        INC     (HL)
017C DD3405     01950        INC     (IX+IND)
017F FD3405     01960        INC     (IY+IND)
0182 3C         01970        INC     A
0183 04         01980        INC     B
0184 03         01990        INC     BC
0185 0C         02000        INC     C
0186 14         02010        INC     D
0187 13         02020        INC     DE
0188 1C         02030        INC     E
0189 24         02040        INC     H
018A 23         02050        INC     HL
018B DD23       02060        INC     IX
018D FD23       02070        INC     IY
018F 2C         02080        INC     L
0190 33         02090        INC     SP
0191 EDAA       02100        IND
0193 EDBA       02110        INDR
0195 EDA2       02120        INI
0197 EDB2       02130        INIR
0199 E9         02140        JP      (HL)
019A DDE9       02150        JP      (IX)
019C FDE9       02160        JP      (IY)
019E DAEEFF     02170        JP      C,NN
01A1 FAEEFF     02180        JP      M,NN
01A4 D2EEFF     02190        JP      NC,NN
01A7 C3EEFF     02200        JP      NN
01AA C2EEFF     02210        JP      NZ,NN
01AD F2EEFF     02220        JP      P,NN
01B0 EAEEFF     02230        JP      PE,NN
01B3 E2EEFF     02240        JP      PO,NN
01B6 CAEEFF     02250        JP      Z,NN
01B9 382E       02260        JR      C,$+DIS
01BB 182E       02270        JR      $+DIS
01BD 302E       02280        JR      NC,$+DIS
01BF 202E       02290        JR      NZ,$+DIS
01C1 282E       02300        JR      Z,$+DIS
01C3 02         02310        LD      (BC),A
01C4 12         02320        LD      (DE),A
01C5 77         02330        LD      (HL),A
01C6 70         02340        LD      (HL),B
01C7 71         02350        LD      (HL),C
01C8 72         02360        LD      (HL),D
01C9 73         02370        LD      (HL),E
01CA 74         02380        LD      (HL),H
01CB 75         02390        LD      (HL),L
01CC 3610       02400        LD      (HL),N
01CE DD7705     02410        LD      (IX+IND),A
01D1 DD7005     02420        LD      (IX+IND),B
01D4 DD7105     02430        LD      (IX+IND),C
01D7 DD7205     02440        LD      (IX+IND),D
01DA DD7305     02450        LD      (IX+IND),E
01DD DD7405     02460        LD      (IX+IND),H
01E0 DD7505     02470        LD      (IX+IND),L
01E3 DD360510   02480        LD      (IX+IND),N
01E7 FD7705     02490        LD      (IY+IND),A
```

```
01EA FD7005     02500          LD       (IY+IND),B
01ED FD7105     02510          LD       (IY+IND),C
01F0 FD7205     02520          LD       (IY+IND),D
01F3 FD7305     02530          LD       (IY+IND),E
01F6 FD7405     02540          LD       (IY+IND),H
01F9 FD7505     02550          LD       (IY+IND),L
01FC FD360510   02560          LD       (IY+IND),N
0200 32EEFF     02570          LD       (NN),A
0203 ED43EEFF   02580          LD       (NN),BC
0207 ED53EEFF   02590          LD       (NN),DE
020B 22EEFF     02600          LD       (NN),HL
020E DD22EEFF   02610          LD       (NN),IX
0212 FD22EEFF   02620          LD       (NN),IY
0216 ED73EEFF   02630          LD       (NN),SP
021A 0A         02640          LD       A,(BC)
021B 1A         02650          LD       A,(DE)
021C 7E         02660          LD       A,(HL)
021D DD7E05     02670          LD       A,(IX+IND)
0220 FD7E05     02680          LD       A,(IY+IND)
0223 3AEEFF     02690          LD       A,(NN)
0226 7F         02700          LD       A,A
0227 78         02710          LD       A,B
0228 79         02720          LD       A,C
0229 7A         02730          LD       A,D
022A 7B         02740          LD       A,E
022B 7C         02750          LD       A,H
022C ED57       02760          LD       A,I
022E 7D         02770          LD       A,L
022F 3E10       02780          LD       A,N
0231 ED5F       02790          LD       A,R
0233 46         02800          LD       B,(HL)
0234 DD4605     02810          LD       B,(IX+IND)
0237 FD4605     02820          LD       B,(IY+IND)
023A 47         02830          LD       B,A
023B 40         02840          LD       B,B
023C 41         02850          LD       B,C
023D 42         02860          LD       B,D
023E 43         02870          LD       B,E
023F 44         02880          LD       B,H
0240 45         02890          LD       B,L
0241 0610       02900          LD       B,N
0243 ED4BEEFF   02910          LD       BC,(NN)
0247 01EEFF     02920          LD       BC,NN
024A 4E         02930          LD       C,(HL)
024B DD4E05     02940          LD       C,(IX+IND)
024E FD4E05     02950          LD       C,(IY+IND)
0251 4F         02960          LD       C,A
0252 48         02970          LD       C,B
0253 49         02980          LD       C,C
0254 4A         02990          LD       C,D
0255 4B         03000          LD       C,E
0256 4C         03010          LD       C,H
0257 4D         03020          LD       C,L
0258 0E10       03030          LD       C,N
025A 56         03040          LD       D,(HL)
025B DD5605     03050          LD       D,(IX+IND)
025E FD5605     03060          LD       D,(IY+IND)
0261 57         03070          LD       D,A
0262 50         03080          LD       D,B
0263 51         03090          LD       D,C
0264 52         03100          LD       D,D
0265 53         03110          LD       D,E
0266 54         03120          LD       D,H
0267 55         03130          LD       D,L
0268 1610       03140          LD       D,N
026A ED5BEEFF   03150          LD       DE,(NN)
026E 11EEFF     03160          LD       DE,NN
0271 5E         03170          LD       E,(HL)
```

```
0272 DD5E05    03180          LD      E,(IX+IND)
0275 FD5E05    03190          LD      E,(IY+IND)
0278 5F        03200          LD      E,A
0279 58        03210          LD      E,B
027A 59        03220          LD      E,C
027B 5A        03230          LD      E,D
027C 5B        03240          LD      E,E
027D 5C        03250          LD      E,H
027E 5D        03260          LD      E,L
027F 1E10      03270          LD      E,N
0281 66        03280          LD      H,(HL)
0282 DD6605    03290          LD      H,(IX+IND)
0285 FD6605    03300          LD      H,(IY+IND)
0288 67        03310          LD      H,A
0289 60        03320          LD      H,B
028A 61        03330          LD      H,C
028B 62        03340          LD      H,D
028C 63        03350          LD      H,E
028D 64        03360          LD      H,H
028E 65        03370          LD      H,L
028F 2610      03380          LD      H,N
0291 2AEEFF    03390          LD      HL,(NN)
0294 21EEFF    03400          LD      HL,NN
0297 ED47      03410          LD      I,A
0299 DD2AEEFF  03420          LD      IX,(NN)
029D DD21EEFF  03430          LD      IX,NN
02A1 FD2AEEFF  03440          LD      IY,(NN)
02A5 FD21EEFF  03450          LD      IY,NN
02A9 6E        03460          LD      L,(HL)
02AA DD6E05    03470          LD      L,(IX+IND)
02AD FD6E05    03480          LD      L,(IY+IND)
02B0 6F        03490          LD      L,A
02B1 68        03500          LD      L,B
02B2 69        03510          LD      L,C
02B3 6A        03520          LD      L,D
02B4 6B        03530          LD      L,E
02B5 6C        03540          LD      L,H
02B6 6D        03550          LD      L,L
02B7 2E10      03560          LD      L,N
02B9 ED4F      03570          LD      R,A
02BB ED7BEEFF  03580          LD      SP,(NN)
02BF F9        03590          LD      SP,HL
02C0 DDF9      03600          LD      SP,IX
02C2 FDF9      03610          LD      SP,IY
02C4 31EEFF    03620          LD      SP,NN
02C7 EDA8      03630          LDD
02C9 EDB8      03640          LDDR
02CB EDA0      03650          LDI
02CD EDB0      03660          LDIR
02CF ED44      03670          NEG
02D1 00        03680          NOP
02D2 B6        03690          OR      (HL)
02D3 DDB605    03700          OR      (IX+IND)
02D6 FDB605    03710          OR      (IY+IND)
02D9 B7        03720          OR      A
02DA B0        03730          OR      B
02DB B1        03740          OR      C
02DC B2        03750          OR      D
02DD B3        03760          OR      E
02DE B4        03770          OR      H
02DF B5        03780          OR      L
02E0 F610      03790          OR      N
02E2 EDBB      03800          OTDR
02E4 EDB3      03810          OTIR
02E6 ED79      03820          OUT     (C),A
02E8 ED41      03830          OUT     (C),B
02EA ED49      03840          OUT     (C),C
02EC ED51      03850          OUT     (C),D
```

```
02EE ED59      03860          OUT     (C),E
02F0 ED61      03870          OUT     (C),H
02F2 ED69      03880          OUT     (C),L
02F4 D310      03890          OUT     (N),A
02F6 EDAB      03900          OUTD
02F8 EDA3      03910          OUTI
02FA F1        03920          POP     AF
02FB C1        03930          POP     BC
02FC D1        03940          POP     DE
02FD E1        03950          POP     HL
02FE DDE1      03960          POP     IX
0300 FDE1      03970          POP     IY
0302 F5        03980          PUSH    AF
0303 C5        03990          PUSH    BC
0304 D5        04000          PUSH    DE
0305 E5        04010          PUSH    HL
0306 DDE5      04020          PUSH    IX
0308 FDE5      04030          PUSH    IY
030A CB86      04040          RES     0,(HL)
030C DDCB0586  04050          RES     0,(IX+IND)
0310 FDCB0586  04060          RES     0,(IY+IND)
0314 CB87      04070          RES     0,A
0316 CB80      04080          RES     0,B
0318 CB81      04090          RES     0,C
031A CB82      04100          RES     0,D
031C CB83      04110          RES     0,E
031E CB84      04120          RES     0,H
0320 CB85      04130          RES     0,L
0322 CB8E      04140          RES     1,(HL)
0324 DDCB058E  04150          RES     1,(IX+IND)
0328 FDCB058E  04160          RES     1,(IY+IND)
032C CB8F      04170          RES     1,A
032E CB88      04180          RES     1,B
0330 CB89      04190          RES     1,C
0332 CB8A      04200          RES     1,D
0334 CB8B      04210          RES     1,E
0336 CB8C      04220          RES     1,H
0338 CB8D      04230          RES     1,L
033A CB96      04240          RES     2,(HL)
033C DDCB0596  04250          RES     2,(IX+IND)
0340 FDCB0596  04260          RES     2,(IY+IND)
0344 CB97      04270          RES     2,A
0346 CB90      04280          RES     2,B
0348 CB91      04290          RES     2,C
034A CB92      04300          RES     2,D
034C CB93      04310          RES     2,E
034E CB94      04320          RES     2,H
0350 CB95      04330          RES     2,L
0352 CB9E      04340          RES     3,(HL)
0354 DDCB059E  04350          RES     3,(IX+IND)
0358 FDCB059E  04360          RES     3,(IY+IND)
035C CB9F      04370          RES     3,A
035E CB98      04380          RES     3,B
0360 CB99      04390          RES     3,C
0362 CB9A      04400          RES     3,D
0364 CB9B      04410          RES     3,E
0366 CB9C      04420          RES     3,H
0368 CB9D      04430          RES     3,L
036A CBA6      04440          RES     4,(HL)
036C DDCB05A6  04450          RES     4,(IX+IND)
0370 FDCB05A6  04460          RES     4,(IY+IND)
0374 CBA7      04470          RES     4,A
0376 CBA0      04480          RES     4,B
0378 CBA1      04490          RES     4,C
037A CBA2      04500          RES     4,D
037C CBA3      04510          RES     4,E
037E CBA4      04520          RES     4,H
0380 CBA5      04530          RES     4,L
```

```
0382 CBAE       04540            RES     5,(HL)
0384 DDCB05AE   04550            RES     5,(IX+IND)
0388 FDCB05AE   04560            RES     5,(IY+IND)
038C CBAF       04570            RES     5,A
038E CBA8       04580            RES     5,B
0390 CBA9       04590            RES     5,C
0392 CBAA       04600            RES     5,D
0394 CBAB       04610            RES     5,E
0396 CBAC       04620            RES     5,H
0398 CBAD       04630            RES     5,L
039A CBB6       04640            RES     6,(HL)
039C DDCB05B6   04650            RES     6,(IX+IND)
03A0 FDCB05B6   04660            RES     6,(IY+IND)
03A4 CBB7       04670            RES     6,A
03A6 CBB0       04680            RES     6,B
03A8 CBB1       04690            RES     6,C
03AA CBB2       04700            RES     6,D
03AC CBB3       04710            RES     6,E
03AE CBB4       04720            RES     6,H
03B0 CBB5       04730            RES     6,L
03B2 CBBE       04740            RES     7,(HL)
03B4 DDCB05BE   04750            RES     7,(IX+IND)
03B8 FDCB05BE   04760            RES     7,(IY+IND)
03BC CBBF       04770            RES     7,A
03BE CBB8       04780            RES     7,B
03C0 CBB9       04790            RES     7,C
03C2 CBBA       04800            RES     7,D
03C4 CBBB       04810            RES     7,E
03C6 CBBC       04820            RES     7,H
03C8 CBBD       04830            RES     7,L
03CA C9         04840            RET
03CB D8         04850            RET     C
03CC F8         04860            RET     M
03CD D0         04870            RET     NC
03CE C0         04880            RET     NZ
03CF F0         04890            RET     P
03D0 E8         04900            RET     PE
03D1 E0         04910            RET     PO
03D2 C8         04920            RET     Z
03D3 ED4D       04930            RETI
03D5 ED45       04940            RETN
03D7 CB16       04950            RL      (HL)
03D9 DDCB0516   04960            RL      (IX+IND)
03DD FDCB0516   04970            RL      (IY+IND)
03E1 CB17       04980            RL      A
03E3 CB10       04990            RL      B
03E5 CB11       05000            RL      C
03E7 CB12       05010            RL      D
03E9 CB13       05020            RL      E
03EB CB14       05030            RL      H
03ED CB15       05040            RL      L
03EF 17         05050            RLA
03F0 CB06       05060            RLC     (HL)
03F2 DDCB0506   05070            RLC     (IX+IND)
03F6 FDCB0506   05080            RLC     (IY+IND)
03FA CB07       05090            RLC     A
03FC CB00       05100            RLC     B
03FE CB01       05110            RLC     C
0400 CB02       05120            RLC     D
0402 CB03       05130            RLC     E
0404 CB04       05140            RLC     H
0406 CB05       05150            RLC     L
0408 07         05160            RLCA
0409 ED6F       05170            RLD
040B CB1E       05180            RR      (HL)
040D DDCB051E   05190            RR      (IX+IND)
0411 FDCB051E   05200            RR      (IY+IND)
0415 CB1F       05210            RR      A
```

```
0417 CB18      05220        RR      B
0419 CB19      05230        RR      C
041B CB1A      05240        RR      D
041D CB1B      05250        RR      E
041F CB1C      05260        RR      H
0421 CB1D      05270        RR      L
0423 1F        05280        RRA
0424 CB0E      05290        RRC     (HL)
0426 DDCB050E  05300        RRC     (IX+IND)
042A FDCB050E  05310        RRC     (IY+IND)
042E CB0F      05320        RRC     A
0430 CB08      05330        RRC     B
0432 CB09      05340        RRC     C
0434 CB0A      05350        RRC     D
0436 CB0B      05360        RRC     E
0438 CB0C      05370        RRC     H
043A CB0D      05380        RRC     L
043C 0F        05390        RRCA
043D ED67      05400        RRD
043F C7        05410        RST     0
0440 D7        05420        RST     10H
0441 DF        05430        RST     18H
0442 E7        05440        RST     20H
0443 EF        05450        RST     28H
0444 F7        05460        RST     30H
0445 FF        05470        RST     38H
0446 CF        05480        RST     8
0447 9E        05490        SBC     A,(HL)
0448 DD9E05    05500        SBC     A,(IX+IND)
044B FD9E05    05510        SBC     A,(IY+IND)
044E 9F        05520        SBC     A,A
044F 98        05530        SBC     A,B
0450 99        05540        SBC     A,C
0451 9A        05550        SBC     A,D
0452 9B        05560        SBC     A,E
0453 9C        05570        SBC     A,H
0454 9D        05580        SBC     A,L
0455 DE10      05590        SBC     A,N
0457 ED42      05600        SBC     HL,BC
0459 ED52      05610        SBC     HL,DE
045B ED62      05620        SBC     HL,HL
045D ED72      05630        SBC     HL,SP
045F 37        05640        SCF
0460 CBC6      05650        SET     0,(HL)
0462 DDCB05C6  05660        SET     0,(IX+IND)
0466 FDCB05C6  05670        SET     0,(IY+IND)
046A CBC7      05680        SET     0,A
046C CBC0      05690        SET     0,B
046E CBC1      05700        SET     0,C
0470 CBC2      05710        SET     0,D
0472 CBC3      05720        SET     0,E
0474 CBC4      05730        SET     0,H
0476 CBC5      05740        SET     0,L
0478 CBCE      05750        SET     1,(HL)
047A DDCB05CE  05760        SET     1,(IX+IND)
047E FDCB05CE  05770        SET     1,(IY+IND)
0482 CBCF      05780        SET     1,A
0484 CBC8      05790        SET     1,B
0486 CBC9      05800        SET     1,C
0488 CBCA      05810        SET     1,D
048A CBCB      05820        SET     1,E
048C CBCC      05830        SET     1,H
048E CBCD      05840        SET     1,L
0490 CBD6      05850        SET     2,(HL)
0492 DDCB05D6  05860        SET     2,(IX+IND)
0496 FDCB05D6  05870        SET     2,(IY+IND)
049A CBD7      05880        SET     2,A
049C CBD0      05890        SET     2,B
```

```
049E CBD1       05900          SET      2,C
04A0 CBD2       05910          SET      2,D
04A2 CBD3       05920          SET      2,E
04A4 CBD4       05930          SET      2,H
04A6 CBD5       05940          SET      2,L
04A8 CBDE       05950          SET      3,(HL)
04AA DDCB05DE 05960            SET      3,(IX+IND)
04AE FDCB05DE 05970            SET      3,(IY+IND)
04B2 CBDF       05980          SET      3,A
04B4 CBD8       05990          SET      3,B
04B6 CBD9       06000          SET      3,C
04B8 CBDA       06010          SET      3,D
04BA CBDB       06020          SET      3,E
04BC CBDC       06030          SET      3,H
04BE CBDD       06040          SET      3,L
04C0 CBE6       06050          SET      4.(HL)
04C2 DDCB05E6 06060            SET      4.(IX+IND)
04C6 FDCB05E6 06070            SET      4,(IY+IND)
04CA CBE7       06080          SET      4,A
04CC CBE0       06090          SET      4,B
04CE CBE1       06100          SET      4,C
04D0 CBE2       06110          SET      4,D
04D2 CBE3       06120          SET      4,E
04D4 CBE4       06130          SET      4,H
04D6 CBE5       06140          SET      4.L
04D8 CBEE       06150          SET      5,(HL)
04DA DDCB05EE 06160            SET      5,(IX+IND)
04DE FDCB05EE 06170            SET      5,(IY+IND)
04E2 CBEF       06180          SET      5.A
04E4 CBE8       06190          SET      5,B
04E6 CBE9       06200          SET      5,C
04E8 CBEA       06210          SET      5,D
04EA CBEB       06220          SET      5,E
04EC CBEC       06230          SET      5,H
04EE CBED       06240          SET      5,L
04F0 CBF6       06250          SET      6,(HL)
04F2 DDCB05F6 06260            SET      6,(IX+IND)
04F6 FDCB05F6 06270            SET      6,(IY+IND)
04FA CBF7       06280          SET      6,A
04FC CBF0       06290          SET      6,B
04FE CBF1       06300          SET      6.C
0500 CBF2       06310          SET      6,D
0502 CBF3       06320          SET      6,E
0504 CBF4       06330          SET      6,H
0506 CBF5       06340          SET      6.L
0508 CBFE       06350          SET      7,(HL)
050A DDCB05FE 06360            SET      7,(IX+IND)
050E FDCB05FE 06370            SET      7,(IY+IND)
0512 CBFF       06380          SET      7.A
0514 CBF8       06390          SET      7,B
0516 CBF9       06400          SET      7.C
0518 CBFA       06410          SET      7,D
051A CBFB       06420          SET      7,E
051C CBFC       06430          SET      7,H
051E CBFD       06440          SET      7,L
0520 CB26       06450          SLA      (HL)
0522 DDCB0526 06460            SLA      (IX+IND)
0526 FDCB0526 06470            SLA      (IY+IND)
052A CB27       06480          SLA      A
052C CB20       06490          SLA      B
052E CB21       06500          SLA      C
0530 CB22       06510          SLA      D
0532 CB23       06520          SLA      E
0534 CB24       06530          SLA      H
0536 CB25       06540          SLA      L
0538 CB2E       06550          SRA      (HL)
053A DDCB052E 06560            SRA      (IX+IND)
053E FDCB052E 06570            SRA      (IY+IND)
```

```
0542 CB2F       06580           SRA     A
0544 CB28       06590           SRA     B
0546 CB29       06600           SRA     C
0548 CB2A       06610           SRA     D
054A CB2B       06620           SRA     E
054C CB2C       06630           SRA     H
054E CB2D       06640           SRA     L
0550 CB3E       06650           SRL     (HL)
0552 DDCB053E   06660           SRL     (IX+IND)
0556 FDCB053E   06670           SRL     (IY+IND)
055A CB3F       06680           SRL     A
055C CB38       06690           SRL     B
055E CB39       06700           SRL     C
0560 CB3A       06710           SRL     D
0562 CB3B       06720           SRL     E
0564 CB3C       06730           SRL     H
0566 CB3D       06740           SRL     L
0568 96         06750           SUB     (HL)
0569 DD9605     06760           SUB     (IX+IND)
056C FD9605     06770           SUB     (IY+IND)
056F 97         06780           SUB     A
0570 90         06790           SUB     B
0571 91         06800           SUB     C
0572 92         06810           SUB     D
0573 93         06820           SUB     E
0574 94         06830           SUB     H
0575 95         06840           SUB     L
0576 D610       06850           SUB     N
0578 AE         06860           XOR     (HL)
0579 DDAE05     06870           XOR     (IX+IND)
057C FDAE05     06880           XOR     (IY+IND)
057F AF         06890           XOR     A
0580 A8         06900           XOR     B
0581 A9         06910           XOR     C
0582 AA         06920           XOR     D
0583 AB         06930           XOR     E
0584 AC         06940           XOR     H
0585 AD         06950           XOR     L
0586 EE10       06960           XOR     N
FFEE            06970 NN        EQU     0FFEEH
0030            06980 DIS       EQU     30H          ; Número de 16 bits
0005            06990 IND       EQU     5            ; Distancia relativa
0010            07000 N         EQU     10H          ; Indice
0000            07010           END                  ; Número de 8 bits
```

Otros libros sobre MICROINFORMATICA

C. Prigmore	**MICROSOFT BASIC** Curso de autoenseñanza para principiantes
G. Ladevie	**La gestión con BASIC** Comercio y pequeña empresa
G. Guérin	**Microinformática de gestión** Alternativas y utilización
A.P. Mullan	**El ordenador en la Educación Básica** Problemática y metodología
D. Daines	**Las bases de datos en la Educación Básica** Utilización y ejemplos
G.W. Orwig/W.S. Hodges	**Programas educacionales para su ordenador personal**
P. Pellier	**Lenguaje máquina del ZX Spectrum** Subrutinas y trucos
T. Hartnell	**Juegos dinámicos para el ZX Spectrum**
R.G. Hurley	**Los Micro Drives del ZX Spectrum** Utilización y aplicaciones
I. Sinclair	**Introducción al Commodore 64**
I. Sinclair	**Lenguaje máquina del Commodore 64**
S. Money	**Gráficos y sonidos para el Commodore 64**
O. Bishop	**Juegos para el Commodore 64**
B. Lloyd	**Introducción al Dragon**
D. Lawrence	**Programas prácticos para el Dragon**
K.S. Brain	**Gráficos y sonidos para el Dragon** Incluye subrutinas en código máquina
K.S. Brain	**Inteligencia artificial en el Dragon**
I. Sinclair	**Lenguaje máquina del Dragon**
M. James/S.M. Gee/K. Ewbank	**Juegos para el Dragon**
V. Apps	**40 juegos educacionales para el Dragon**

Así se empieza
Introducción a los ordenadores
Peter Lafferty

204 páginas, de 20 × 14 cm, con más de 100 ilustraciones a dos colores

Indice. Introducción. 1 ¿Qué es un ordenador doméstico? 2 Cómo utilizar su ordenador. 3 ¿Qué puede hacer usted con un ordenador? 4 Cómo escribir sus propios programas. 5 Cómo funcionan los ordenadores. 6 Ampliación del sistema. 7 Elección del ordenador. 8 Hacia el futuro. Apéndice 1. Apéndice 2. Glosario de terminología de ordenadores. Resumen de ordenadores. Bibliografía. Club de usuarios. Indice analítico.

Primeros pasos en BASIC
Susan Curran - Ray Curnow

208 páginas, de 20 × 14 cm, con más de 60 ilustraciones a dos colores

Indice. Introducción. 1 Cómo escribir en la pantalla. 2 Nuestros primeros programas. 3 Introducción de variables. 4 Bucles y ramificaciones. 5 Edición y corrección de errores. 6 Cómo manejar los datos. 7 Cómo escribir programas más largos. 8 Los siguientes pasos. Apéndice 1. Apéndice 2. Apéndice 3. Respuestas a las preguntas. Indice analítico.

El estudiante y el ordenador
Aplicaciones a la enseñanza
Susan Curran - Ray Curnow

168 páginas, de 20 × 14 cm, con más de 40 ilustraciones a dos colores

Indice. Introducción. 1 El ordenador como una ayuda para aprender. 2 Ordenadores para los niños. 3 Programas de recursos. 4 Cómo comprar software. 5 Hardware para la educación. 6 Algunos programas para que usted los pruebe. Apéndice 1. Apéndice 2. Bibliografía. Clubs de usuarios. Indice analítico.

Juegos, imágenes y sonidos
Susan Curran - Ray Curnow

168 páginas, de 20 × 14 cm, con más de 50 ilustraciones a dos colores

Indice. Introducción. 1 Resumen histórico de los juegos por ordenador. 2 Tipos de juegos para ordenador. 3 Gráficos por ordenador. 4 Generación de sonidos mediante su ordenador. 5 Hardware del ordenador. 6 Cómo escribir programas de juegos. 7 Algunos programas que usted puede probar. 8 Compra de programas. Glosario. Indice analítico.

GG

Franquear
como
Tarjeta
Postal

Editorial Gustavo Gili, S.A.
Apartado de Correos 35.149

08080 Barcelona (España)

SOLICITUD GRATUITA DE CATÁLOGOS

Estimado lector:

Le quedamos muy agradecidos por adquirir este libro, el cual deseamos responda completamente a sus necesidades.
Al devolvernos esta tarjeta solicitando catálogos, le rogamos nos preste su colaboración respondiendo a las siguientes preguntas. Muchas gracias.

¿DE QUÉ OBRA HA RETIRADO ESTA TARJETA? ....

....

¿CÓMO CONOCIÓ ESTE LIBRO?

☐ Reseña crítica ☐ Anuncio prensa ☐ Escaparate ☐ Folleto
☐ Recomendación personal ☐ Aconsejado por el profesor
☐ ....

¿DÓNDE ADQUIRIÓ ESTA OBRA?

☐ Librería ☐ Vendedor visitador
☐ Feria de Libros ☐ .... ☐ Directamente de la Editorial

Apellidos ....

Nombre ....

Profesión ....

Estudiante de ....

.... Curso ....

Dirección particular ....

Población ....

Provincia .... D.P. ....

.... País ....

Deseo recibir gratuitamente, a vuelta de correo, el catálogo de sus publicaciones, así como información periódica de las novedades que vayan publicando sobre las materias que les señalo:

☐ **Arquitectura. Construcción. Urbanismo**
☐ **Ingeniería general**
☐ **Mecánica**
☐ **Plásticos**
☐ **Electricidad**
☐ **Electrónica-Informática**
☐ **Diseño. Dibujo**
☐ **Tecnología y Sociedad**
☐ **Fotografía. Cine. Teatro. Televisión**
☐ **Comunicación. Mass Media**
☐ **Arte. Estética**
☐ **Literatura. Diccionarios**
☐ **Decoración. Muebles**
☐ **Jardinería**
☐ **Obras de arte numeradas**
☐ **Interés general**

Esta obra presenta de manera progresiva el ensamblador del microprocesador Z 80 y su aplicación sobre el ZX SPECTRUM.

Está destinada a aquellas personas que desean saber más sobre el funcionamiento interno de su microordenador para sacarle el mejor partido, según sus posibilidades, y aumentar la velocidad de ejecución de los programas. El ensamblador es, en efecto, el lenguaje más rápido que pueda utilizarse sobre un microordenador. Funciona aproximadamente 100 veces más rápido que el BASIC y es el único que permite la ejecución de juegos de acción rápida de alto nivel.

Además de las instrucciones abundantemente detalladas del Z 80, este libro suministra una serie de subprogramas que serán excelentes ejemplos para los principiantes, a los cuales permitirá fácilmente la ejecución de complejos programas en ensamblador. Particularmente explica cómo escribir o dibujar en la pantalla o en la impresora, programar la salida sonora o la salida de cassette y detectar las teclas pulsadas en el teclado.

Es un útil indispensable para todo aquel que desee programar en ensamblador sobre el ZX SPECTRUM.

Editorial Gustavo Gili, S. A.
Rosellón, 87-89
08029 Barcelona